Package 'rddtools'

October 22, 2026

Version 2.0.3

Date 2026-10-22

Title Toolbox for Regression Discontinuity Design ('RDD')

Description Set of functions for Regression Discontinuity Design ('RDD'), for data visualisation, estimation and testing.

Imports KernSmooth, ggplot2, sandwich, lmtest, Formula, locpol, methods, rdrobust

Depends AER, np

Suggests stats4, car, knitr, rmarkdown, testthat

License GPL (>= 3)

URL https://github.com/bquast/rddtools

BugReports https://github.com/bquast/rddtools/issues

VignetteBuilder knitr

Encoding UTF-8

Collate 'as.npreg.R' 'bw_cct_estim.R' 'bw_cct_plot.R' 'bw_ik.R' 'bw_rot.R' 'clusterInf.R' 'covarTests.R' 'dens_test.R' 'gen_mc_ik.R' 'get_methods.R' 'model.matrix.rdd.R' 'placebo.R' 'plotBin.R' 'plotSensi.R' 'qplot_experim.R' 'rdd_coef.R' 'rdd_data.R' 'rdd_data_methods.R' 'rdd_pkg_old_kernelwts.R' 'rdd_pkg_old_DCdensity.R' 'rdd_pred.R' 'rddtools.R' 'reg_gen.R' 'reg_lm.R' 'reg_np.R' 'var_estim.R' 'various_code.R' 'waldci.R'

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Matthieu Stigler [aut, cre] (ORCID: <https://orcid.org/0000-0002-6802-4290>), Bastiaan Quast [aut] (ORCID: <https://orcid.org/0000-0002-2951-3577>)

Maintainer Matthieu Stigler <Matthieu.Stigler@gmail.com>

Repository CRAN

Date/Publication 2026-10-22 17:10:02 UTC

Contents

rddtools-package . . . 2
as.lm . . . 3
as.npregbw . . . 4
clusterInf . . . 5
covarTest_dis . . . 6
covarTest_mean . . . 7
dens_test . . . 9
gen_mc_ik . . . 10
house . . . 11
indh . . . 12
plot.rdd_data . . . 12
plotBin . . . 14
plotPlacebo . . . 15
plotSensi . . . 17
rdd_bw_cct_estim . . . 19
rdd_bw_cct_plot . . . 20
rdd_bw_ik . . . 21
rdd_bw_rsw . . . 22
rdd_coef . . . 23
rdd_data . . . 23
rdd_gen_reg . . . 25
rdd_pred . . . 26
rdd_reg_lm . . . 28
rdd_reg_np . . . 30
rot_bw . . . 31
STAR_MHE . . . 32
vcovCluster . . . 33
waldci . . . 34

Index **35**

`rddtools-package` *Regression Discontinuity Design*

Description

Set of functions for Regression Discontinuity Design ('RDD'), for data visualisation, estimation and testing.

Author(s)

Maintainer: Matthieu Stigler <Matthieu.Stigler@gmail.com> (ORCID)

Authors:

- Matthieu Stigler <Matthieu.Stigler@gmail.com> (ORCID)
- Bastiaan Quast <bquast@gmail.com> (ORCID)

See Also

Useful links:

- https://github.com/bquast/rddtools
- Report bugs at https://github.com/bquast/rddtools/issues

`as.lm` *Convert a rdd object to lm*

Description

Convert a rdd object to lm

Usage

```
as.lm(x)
```

Arguments

`x`	An object to convert to lm

Value

An object of class `lm`

See Also

`as.npreg` which converts `rdd_reg` objects into `npreg` from package `np`.

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
reg_para <- rdd_reg_lm(rdd_object=house_rdd)
reg_para_lm <- as.lm(reg_para)
reg_para_lm
plot(reg_para_lm, which=4)
```

as.npregbw *Convert an rdd_reg object to a* npreg *object*

Description

Convert an rdd_object to a non-parametric regression npreg from package np

Usage

```
as.npregbw(x, ...)

as.npreg(x, ...)
```

Arguments

x	Object of class rdd_reg created by rdd_reg_np or rdd_reg_lm
...	Further arguments passed to the npregbw or npreg

Details

This function converts an rdd_reg object into an npreg object from package np Note that the output won't be the same, since npreg does not offer a triangular kernel, but a Gaussian or Epanechinkov one. Another reason why estimates might differ slightly is that npreg implements a multivariate kernel, while rdd_reg proceeds as if the kernel was univariate. A simple solution to make the multivariate kernel similar to the univariate one is to set the bandwidth for x and Dx to a large number, so that they converge towards a constant, and one obtains back the univariate kernel.

Value

An object of class npreg or npregbw

See Also

as.lm which converts rdd_reg objects into lm.

Examples

```
# Estimate ususal rdd_reg:
 data(house)
 house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
 reg_nonpara <- rdd_reg_np(rdd_object=house_rdd)

## Convert to npreg:
 reg_nonpara_np <- as.npreg(reg_nonpara)
 reg_nonpara_np
 rdd_coef(reg_nonpara_np, allCo=TRUE, allInfo=TRUE)

## Compare with result obtained with a Gaussian kernel:
```

```
bw_lm <- dnorm(house_rdd$x, sd=rddtools:::getBW(reg_nonpara))
reg_nonpara_gaus <- rdd_reg_lm(rdd_object=house_rdd, w=bw_lm)
all.equal(rdd_coef(reg_nonpara_gaus),rdd_coef(reg_nonpara_np))
```

`clusterInf` *Post-inference for clustered data*

Description

Correct standard-errors to account for clustered data, doing either a degrees of freedom correction or using a heteroskedasticidty-cluster robust covariance matrix possibly on the range specified by bandwidth

Usage

```
clusterInf(object, clusterVar, vcov. = NULL, type = c("df-adj", "HC"), ...)
```

Arguments

`object`	Object of class lm, from which rdd_reg also inherits.
`clusterVar`	The variable containing the cluster attributions.
`vcov.`	Specific covariance function to pass to coeftest. See help of sandwich
`type`	The type of cluster correction to use: either the degrees of freedom, or a HC matrix.
`...`	Further arguments passed to coeftest

Value

The output of the coeftest function, which is itself of class `coeftest`

References

Wooldridge (2003) Cluster-sample methods in applied econometrics. *AmericanEconomic Review*, 93, p. 133-138

See Also

`vcovCluster`, which implements the cluster-robust covariance matrix estimator used by `cluserInf`

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
reg_para <- rdd_reg_lm(rdd_object=house_rdd)

# here we just generate randomly a cluster variable:
nlet <- sort(c(outer(letters, letters, paste, sep='')))
clusRandom <- sample(nlet[1:60], size=nrow(house_rdd), replace=TRUE)
```

```
# now do post-inference:
clusterInf(reg_para, clusterVar=clusRandom)
clusterInf(reg_para, clusterVar=clusRandom, type='HC')
```

`covarTest_dis` *Testing for balanced covariates: equality of distribution*

Description

Tests equality of distribution with a Kolmogorov-Smirnov for each covariates, between the two full groups or around the discontinuity threshold

Usage

```
covarTest_dis(
  object,
  bw,
  exact = NULL,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
)

## S3 method for class 'rdd_data'
covarTest_dis(
  object,
  bw = NULL,
  exact = FALSE,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
)

## S3 method for class 'rdd_reg'
covarTest_dis(
  object,
  bw = NULL,
  exact = FALSE,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
)
```

Arguments

`object`	object of class rdd_data
`bw`	a bandwidth
`exact`	Argument of the `ks.test` function: NULL or a logical indicating whether an exact p-value should be computed.
`p.adjust`	Whether to adjust the p-values for multiple testing. Uses the `p.adjust` function

Value

A data frame with, for each covariate, the K-S statistic and its p-value.

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

See Also

covarTest_mean for the t-test of equality of means

Examples

```
data(house)

## Add randomly generated covariates
set.seed(123)
n_Lee <- nrow(house)
Z <- data.frame(z1 = rnorm(n_Lee, sd=2),
                z2 = rnorm(n_Lee, mean = ifelse(house<0, 5, 8)),
                z3 = sample(letters, size = n_Lee, replace = TRUE))
house_rdd_Z <- rdd_data(y = house$y, x = house$x, covar = Z, cutpoint = 0)

## Kolmogorov-Smirnov test of equality in distribution:
covarTest_dis(house_rdd_Z, bw=0.3)

## Can also use function covarTest_dis() for a t-test for equality of means around cutoff:
covarTest_mean(house_rdd_Z, bw=0.3)
## covarTest_dis works also on regression outputs (bw will be taken from the model)
reg_nonpara <- rdd_reg_np(rdd_object=house_rdd_Z)
covarTest_dis(reg_nonpara)
```

---

covarTest_mean *Testing for balanced covariates: equality of means with t-test*

Description

Tests equality of means by a t-test for each covariate, between the two full groups or around the discontinuity threshold

Usage

```
covarTest_mean(
  object,
  bw = NULL,
  paired = FALSE,
  var.equal = FALSE,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
```

```
)

## S3 method for class 'rdd_data'
covarTest_mean(
  object,
  bw = NULL,
  paired = FALSE,
  var.equal = FALSE,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
)

## S3 method for class 'rdd_reg'
covarTest_mean(
  object,
  bw = NULL,
  paired = FALSE,
  var.equal = FALSE,
  p.adjust = c("none", "holm", "BH", "BY", "hochberg", "hommel", "bonferroni")
)
```

Arguments

object	object of class rdd_data
bw	a bandwidth
paired	Argument of the t.test function: logical indicating whether you want paired t-tests.
var.equal	Argument of the t.test function: logical variable indicating whether to treat the two variances as being equal
p.adjust	Whether to adjust the p-values for multiple testing. Uses the p.adjust function

Value

A data frame with, for each covariate, the mean on each size, the difference, t-stat and ts p-value.

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

See Also

covarTest_dis for the Kolmogorov-Smirnov test of equality of distribution

Examples

```
data(house)

## Add randomly generated covariates
set.seed(123)
n_Lee <- nrow(house)
```

```
Z <- data.frame(z1 = rnorm(n_Lee, sd=2),
                z2 = rnorm(n_Lee, mean = ifelse(house<0, 5, 8)),
                z3 = sample(letters, size = n_Lee, replace = TRUE))
house_rdd_Z <- rdd_data(y = house$y, x = house$x, covar = Z, cutpoint = 0)

## test for equality of means around cutoff:
covarTest_mean(house_rdd_Z, bw=0.3)

## Can also use function covarTest_dis() for Kolmogorov-Smirnov test:
covarTest_dis(house_rdd_Z, bw=0.3)

## covarTest_mean works also on regression outputs (bw will be taken from the model)
reg_nonpara <- rdd_reg_np(rdd_object=house_rdd_Z)
covarTest_mean(reg_nonpara)
```

`dens_test` *McCrary Sorting Test*

Description

This calls the original `DCdensity` function which was in the package `rdd` by Drew Dimmery, which has been archived and is now internally stored in the Rddtools package.

Usage

```
dens_test(rdd_object, bin = NULL, bw = NULL, plot = TRUE, ...)
```

Arguments

`rdd_object`	object of class rdd_data
`bin`	the binwidth (defaults to `2*sd(runvar)*length(runvar)^(-.5)`)
`bw`	the bandwidth to use (by default uses bandwidth selection calculation from McCrary (2008))
`plot`	Whether to return a plot. Logical, default to TRUE.
`...`	Further arguments passed to the unexported `DCdensity` function.

Details

Run the McCracy test for manipulation of the forcing variable

References

McCrary, Justin. (2008) "Manipulation of the running variable in the regression discontinuity design: A density test," *Journal of Econometrics*. 142(2): 698-714. doi:10.1016/j.jeconom.2007.05.005

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
dens_test(house_rdd)
```

---

gen_mc_ik *Generate Monte Carlo simulations of Imbens and Kalyanaraman*

Description

Generate the simulations reported in Imbens and Kalyanaraman (2012)

Usage

```
gen_mc_ik(
  n = 200,
  version = 1,
  sd = 0.1295,
  output = c("data.frame", "rdd_data"),
  size
)
```

Arguments

n	The size of sampel to generate
version	The MC version of Imbens and Kalnayaraman (between 1 and 4).
sd	The standard deviation of the error term.
output	Whether to return a data-frame, or already a rdd_data
size	The size of the effect, this depends on the specific version, defaults are as in ik: 0.04, NULL, 0.1, 0.1

Value

An data frame with x and y variables.

Examples

```
mc1_dat <- gen_mc_ik()
MC1_rdd <- rdd_data(y=mc1_dat$y, x=mc1_dat$x, cutpoint=0)

## Use np regression:
reg_nonpara <- rdd_reg_np(rdd_object=MC1_rdd)
reg_nonpara

# Represent the curves:
plotCu <- function(version=1, xlim=c(-0.1,0.1)){
  res <- gen_mc_ik(sd=0.0000001, n=1000, version=version)
```

```
  res <- res[order(res$x),]
  ylim <- range(subset(res, x>=min(xlim) & x<=max(xlim), 'y'))
  plot(res, type='l', xlim=xlim, ylim=ylim, main=paste('DGP', version))
  abline(v=0)
  xCut <- res[which(res$x==min(res$x[res$x>=0]))+c(0,-1),]
  points(xCut, col=2)
}
layout(matrix(1:4,2, byrow=TRUE))
plotCu(version=1)
plotCu(version=2)
plotCu(version=3)
plotCu(version=4)
layout(matrix(1))
```

---

house *Dataset used in Lee (2008)*

Description

Randomized experiments from non-random selection in U.S. House elections

Dataset described used in Imbens and Kalyamaran (2012), and probably the same dataset used in Lee (2008),

Format

A data frame with 6558 observations and two variables:

x Vote at election t-1

y Vote at election t

Source

Guido Imbens' former Harvard webpage, archived at https://web.archive.org/web/20240811075722/https://scholar.harvard.edu/imbens/scholar_software/regression-discontinuity

References

Imbens, Guido and Karthik Kalyanaraman. (2012) 'Optimal Bandwidth Choice for the regression discontinuity estimator,' Review of Economic Studies (2012) 79, 933-959. doi:10.1093/restud/rdr043

Lee, D. (2008) Randomized experiments from non-random selection in U.S. House elections, *Journal of Econometrics*, 142, 675-697. doi:10.1016/j.jeconom.2007.05.004

Examples

```
data(house)
rdd_house <- rdd_data(x=x, y=y, data=house, cutpoint=0)
summary(rdd_house)
plot(rdd_house)
```

indh *INDH data set*

Description

Data from the Initiative Nationale du Development Humaine, collected as the part of the SNSF project "Development Aid and Social Dynamics"

Format

A data frame with two variables with 720 observations each

References

Arcand, Rieger, and Nguyen (2015) 'Development Aid and Social Dyanmics Data Set'

Examples

```
# load the data
data(indh)

# construct rdd_data frame
rdd_dat_indh <- rdd_data(y=choice_pg, x=poverty, data=indh, cutpoint=30)

# inspect data frame
summary(rdd_dat_indh)

# perform non-parametric regression
( reg_np_indh <- rdd_reg_np(rdd_dat_indh) )
```

plot.rdd_data *Plot rdd_data*

Description

Binned plot of the forcing and outcome variable

Usage

```
## S3 method for class 'rdd_data'
plot(
  x,
  h = NULL,
  nbins = NULL,
  xlim = range(object$x, na.rm = TRUE),
  cex = 0.7,
  nplot = 1,
```

```
  device = c("base", "ggplot"),
  ...
)
```

Arguments

x	Object of class rdd_data
h	The binwidth parameter (note this differs from the bandwidth parameter!)
nbins	Alternative to h, the total number of bins in the plot.
xlim	The range of the x data
cex	Size of the points, see par
nplot	Number of plot to draw
device	Type of device used. Currently not used.
...	Further arguments passed to the plot function.

Details

Produces a simple binned plot averaging values within each interval. The length of the intervals is specified with the argument h, specifying the whole binwidth (contrary to the usual bandwidth argument, that gives half of the length of the kernel window. When no bandwidth is given, the bandwidth of Ruppert et al is used, see rdd_bw_rsw.

Value

A plot

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
plot(house_rdd)

## Specify manually the bandwidth:
plot(house_rdd, h=0.2)

## Show three plots with different bandwidth:
plot(house_rdd, h=c(0.2,0.3,0.4), nplot=3)

## Specify instead of the bandwidth, the final number of bins:
plot(house_rdd, nbins=22)

## If the specified number of bins is odd, the larger number is given to side with largest range
plot(house_rdd, nbins=21)
```

plotBin *Bin plotting*

Description

Do a 'scatterplot bin smoothing'

Usage

```
plotBin(
  x,
  y,
  h = NULL,
  nbins = NULL,
  cutpoint = 0,
  plot = TRUE,
  type = c("value", "number"),
  xlim = range(x, na.rm = TRUE),
  cex = 0.9,
  main = NULL,
  xlab,
  ylab,
  ...
)
```

Arguments

x	Forcing variable
y	Output
h	the bandwidth (defaults to 2*sd(runvar)*length(runvar)^(-.5))
nbins	number of Bins
cutpoint	Cutpoint
plot	Logical. Whether to plot or only returned silently
type	Whether returns the y averages, or the x frequencies
xlim, cex, main, xlab, ylab	Usual parameters passed to plot(), see par
...	further arguments passed to plot.

Value

Returns silently values

References

McCrary, Justin.

plotPlacebo *Draw a (density) plot of placebo tests*

Description

Draw a plot of placebo tests, estimating the impact on fake cutpoints

Usage

```
plotPlacebo(
  object,
  device = c("ggplot", "base"),
  output = c("data", "ggplot"),
  ...
)

## S3 method for class 'rdd_reg'
plotPlacebo(
  object,
  device = c("ggplot", "base"),
  output = c("data", "ggplot"),
  from = 0.25,
  to = 0.75,
  by = 0.1,
  level = 0.95,
  same_bw = FALSE,
  vcov. = NULL,
  plot = TRUE,
  ...
)

plotPlaceboDens(
  object,
  device = c("ggplot", "base"),
  output = c("data", "ggplot"),
  ...
)

## S3 method for class 'rdd_reg'
plotPlaceboDens(
  object,
  device = c("ggplot", "base"),
  output = c("data", "ggplot"),
  from = 0.25,
  to = 0.75,
  by = 0.1,
  level = 0.95,
```

```
  same_bw = FALSE,
  vcov. = NULL,
  ...
)

computePlacebo(
  object,
  from = 0.25,
  to = 0.75,
  by = 0.1,
  level = 0.95,
  same_bw = FALSE,
  vcov. = NULL
)
```

Arguments

object	the output of an RDD regression
device	Whether to draw a base or a ggplot graph.
output	Whether to return (invisibly) the data frame containing the bandwidths and corresponding estimates, or the ggplot object
from	Starting point of the fake cutpoints sequence. Refers ot the quantile of each side of the true cutpoint
to	Ending point of the fake cutpoints sequence. Refers ot the quantile of each side of the true cutpoint
by	Increments of the from-to sequence
level	Level of the confidence interval shown
same_bw	Whether to re-estimate the bandwidth at each point
vcov.	Specific covariance function to pass to coeftest. See help of package sandwich.
plot	Whether to actually plot the data.
...	Further arguments passed to specific methods.

Value

A data frame containing the cutpoints, their corresponding estimates and confidence intervals.

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
reg_nonpara <- rdd_reg_np(rdd_object=house_rdd)
plotPlacebo(reg_nonpara)
```

```
# Use with another vcov function; cluster case
reg_nonpara_lminf <- rdd_reg_np(rdd_object=house_rdd, inference='lm')
# need to be a function applied to updated object!
vc <- function(x) vcovCluster(x, clusterVar=model.frame(x)$x)
plotPlacebo(reg_nonpara_lminf, vcov. = vc)
```

`plotSensi` *Plot the sensitivity to the bandwidth*

Description

Draw a plot showing the LATE estimates depending on multiple bandwidths

Usage

```
plotSensi(
  rdd_regobject,
  from,
  to,
  by = 0.01,
  level = 0.95,
  output = c("data", "ggplot"),
  plot = TRUE,
  ...
)

## S3 method for class 'rdd_reg_np'
plotSensi(
  rdd_regobject,
  from,
  to,
  by = 0.05,
  level = 0.95,
  output = c("data", "ggplot"),
  plot = TRUE,
  device = c("ggplot", "base"),
  vcov. = NULL,
  ...
)

## S3 method for class 'rdd_reg_lm'
plotSensi(
  rdd_regobject,
  from,
  to,
  by = 0.05,
  level = 0.95,
```

```
  output = c("data", "ggplot"),
  plot = TRUE,
  order,
  type = c("colour", "facet"),
  ...
)
```

Arguments

rdd_regobject	object of a RDD regression, from either rdd_reg_lm or rdd_reg_np
from	First bandwidth point. Default value is max(1e-3, bw-0.1)
to	Last bandwidth point. Default value is bw+0.1
by	Increments in the from to sequence
level	Level of the confidence interval
output	Whether to return (invisibly) the data frame containing the bandwidths and corresponding estimates, or the ggplot object
plot	Whether to actually plot the data.
device	Whether to draw a base or a ggplot graph.
vcov.	Specific covariance function to pass to coeftest. See help of package sandwich
order	For parametric models (from rdd_reg_lm), the order of the polynomial.
type	For parametric models (from rdd_reg_lm) whether different orders are represented as different colour or as different facets.
...	Further arguments passed to specific methods

Value

A data frame containing the bandwidths and corresponding estimates and confidence intervals.

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

Examples

```
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)

#Non-parametric estimate
bw_ik <- rdd_bw_ik(house_rdd)
reg_nonpara <- rdd_reg_np(rdd_object=house_rdd, bw=bw_ik)
plotSensi(reg_nonpara)
plotSensi(reg_nonpara, device='base')

#Parametric estimate:
reg_para_ik <- rdd_reg_lm(rdd_object=house_rdd, order=4, bw=bw_ik)
plotSensi(reg_para_ik)
plotSensi(reg_para_ik, type='facet')
```

`rdd_bw_cct_estim` *Bandwidth selection for Regression Discontinuity estimators, CTT 2014*

Description

Simple wrapper of the Calonico-Cattaneo-Titiunik (2014) bandwidth selection procedures for RDD estimators `rdbwselect`.

Usage

```
rdd_bw_cct_estim(
  rdd_object,
 method = c("mserd", "msetwo", "msesum", "msecomb1", "msecomb2", "cerrd", "certwo",
    "cersum", "cercomb1"),
  kernel = c("Triangular", "Uniform", "Epanechnikov"),
  ...
)
```

Arguments

`rdd_object`	of class rdd_data created by `rdd_data`
`method`	The type of method used. See
`kernel`	The type of kernel used: either `Triangular`, `Uniform` or `Epanechnikov`.
`...`	further arguments passed to `rdbwselect`.

Value

See documentation of `rdbwselect`

Author(s)

Original code written by Calonico, Cattaneo, Farrell and Titiuni, see `rdbwselect`

References

Calonico, S., M. D. Cattaneo, and R. Titiunik. 2014a. Robust Nonparametric Confidence Intervals for Regression-Discontinuity Designs. Econometrica 82(6): 2295-2326. https://www.tandfonline.com/doi/abs/10.1080/01621459.2015.1017578.

See Also

`rdd_bw_ik` Local RDD bandwidth selector using the plug-in method of Imbens and Kalyanaraman (2012)

Examples

```
data(house)
rd<- rdd_data(x=house$x, y=house$y, cutpoint=0)
rdd_bw_cct_estim(rd)
```

`rdd_bw_cct_plot` *Bandwidth selection for Regression Discontinuity visualisation, CTT 2015*

Description

Simple wrapper of the Calonico-Cattaneo-Titiunik (2015) bandwidth selection procedures for RDD visualisation `rdplot`.

Usage

```
rdd_bw_cct_plot(
  rdd_object,
  method = c("esmv", "es", "espr", "esmvpr", "qs", "qspr", "qsmv", "qsmvpr"),
  ...
)
```

Arguments

`rdd_object`	of class rdd_data created by `rdd_data`
`method`	The type of method used. See `rdplot`. Default is `esmv`, the variance mimicking evenly-spaced method.
`...`	further arguments passed to `rdplot`.

Value

See documentation of `rdplot`

Author(s)

Original code written by Calonico, Cattaneo, Farrell and Titiuni, see `rdplot`

References

Calonico, S., M. D. Cattaneo, and R. Titiunik. 2015a. Optimal Data-Driven Regression Discontinuity Plots. Journal of the American Statistical Association 110(512): 1753-1769. https://www.tandfonline.com/doi/abs/10.1080/01621459.2015.1017578.

See Also

`rdd_bw_ik` Local RDD bandwidth selector using the plug-in method of Imbens and Kalyanaraman (2012)

Examples

```
data(house)
rd<- rdd_data(x=house$x, y=house$y, cutpoint=0)
rdd_bw_cct_plot(rd)
```

---

rdd_bw_ik *Imbens-Kalyanaraman Optimal Bandwidth Calculation*

Description

Imbens-Kalyanaraman optimal bandwidth for local linear regression in Regression discontinuity designs.

Usage

```
rdd_bw_ik(rdd_object, kernel = c("Triangular", "Uniform", "Normal"))
```

Arguments

rdd_object	of class rdd_data created by rdd_data
kernel	The type of kernel used: either triangular or uniform.

Value

The optimal bandwidth

Author(s)

Matthieu Stigler <<Matthieu.Stigler@gmail.com>>

References

Imbens, Guido and Karthik Kalyanaraman. (2012) 'Optimal Bandwidth Choice for the regression discontinuity estimator,' Review of Economic Studies (2012) 79, 933-959

See Also

rdd_bw_rsw Global bandwidth selector of Ruppert, Sheather and Wand (1995)

Examples

```
data(house)
rd<- rdd_data(x=house$x, y=house$y, cutpoint=0)
rdd_bw_ik(rd)
```

rdd_bw_rsw *Global bandwidth selector of Ruppert, Sheather and Wand (1995) from package* **KernSmooth**

Description

Uses the global bandwidth selector of Ruppert, Sheather and Wand (1995) either to the whole function, or to the functions below and above the cutpoint.

Usage

```
rdd_bw_rsw(object, type = c("global", "sided"))
```

Arguments

object	object of class rdd_data created by rdd_data
type	Whether to choose a global bandwidth for the whole function (global) or for each side (sided)

Value

One (or two for sided) bandwidth value.

References

See dpill

See Also

rdd_bw_ik Local RDD bandwidth selector using the plug-in method of Imbens and Kalyanaraman (2012)

Examples

```
data(house)
rd<- rdd_data(x=house$x, y=house$y, cutpoint=0)
rdd_bw_rsw(rd)
```

rdd_coef *RDD coefficient*

Description

Function to access the RDD coefficient in the various regressions

Usage

```
rdd_coef(object, allInfo = FALSE, allCo = FALSE, ...)

## Default S3 method:
rdd_coef(object, allInfo = FALSE, allCo = FALSE, ...)

## S3 method for class 'rdd_reg_np'
rdd_coef(object, allInfo = FALSE, allCo = FALSE, ...)
```

Arguments

object	A RDD regression object
allInfo	whether to return just the coefficients (allInfo=FALSE) or also the se/t stat/pval.
allCo	Whether to give only the RDD coefficient (allCo=FALSE) or all coefficients
...	Further arguments passed to/from specific methods

Value

Either a numeric value of the RDD coefficient estimate, or a data frame with the estimate, its standard value, t test and p-value and

rdd_data *Construct rdd_data*

Description

Construct the base RDD object, containing x, y and the cutpoint, eventuallay covariates.

Usage

```
rdd_data(y, x, covar, cutpoint, z, labels, data)
```

Arguments

y	Output
x	Forcing variable
covar	Exogeneous variables
cutpoint	Cutpoint
z	Assignment variable for the fuzzy case. Should be 0/1 or TRUE/FALSE variable.
labels	Additional labels to provide as list (with entries x, y, and eventually vector covar). Unused currently.
data	A data-frame for the x and y variables. If this is provided, the column names can be entered directly for argument x, y and covar. For covar, should be a character vector.

Details

Arguments x, y (and eventually covar) can be either given as:

- vectors (eventually data-frame for covar)
- quote/character when data is also provided. For multiple covar, use a vector of characters

Value

Object of class rdd_data, inheriting from data.frame

Author(s)

Matthieu Stigler <Matthieu.Stigler@gmail.com>

Examples

```
data(house)
rd <- rdd_data(x=house$x, y=house$y, cutpoint=0)
rd2 <- rdd_data(x=x, y=y, data=house, cutpoint=0)

# The print() function is the same as the print.data.frame:
rd

# The summary() and plot() function are specific to rdd_data
summary(rd)
plot(rd)

# for the fuzzy case, you need to specify the assignment variable z:
rd_dat_fakefuzzy <- rdd_data(x=house$x, y=house$y,
                             z=ifelse(house$x>0+rnorm(nrow(house), sd=0.05),1,0),
                             cutpoint=0)
summary(rd_dat_fakefuzzy)
```

rdd_gen_reg *General polynomial estimator of the regression discontinuity*

Description

Compute RDD estimate allowing a locally kernel weighted version of any estimation function possibly on the range specified by bandwidth

Usage

```
rdd_gen_reg(
  rdd_object,
  fun = glm,
  covariates = NULL,
  order = 1,
  bw = NULL,
  slope = c("separate", "same"),
 covar.opt = list(strategy = c("include", "residual"), slope = c("same", "separate"), bw
    = NULL),
  weights,
  ...
)
```

Arguments

rdd_object	Object of class rdd_data created by rdd_data
fun	The function to estimate the parameters
covariates	Formula to include covariates
order	Order of the polynomial regression.
bw	A bandwidth to specify the subset on which the kernel weighted regression is estimated
slope	Whether slopes should be different on left or right (separate), or the same.
covar.opt	Options for the inclusion of covariates. Way to include covariates, either in the main regression (include) or as regressors of y in a first step (residual).
weights	Optional weights to pass to the lm function. Note this cannot be entered together with bw
...	Further arguments passed to fun. See the example.

Details

This function allows the user to use a custom estimating function, instead of the traditional lm(). It is assumed that the custom funciton has following behaviour:

1. A formula interface, together with a data argument
2. A weight argument

3. A coef(summary(x)) returning a data-frame containing a column Estimate

Note that for the last requirement, this can be accomodated by writing a specific rdd_coef function for the class of the object returned by fun.

Value

An object of class rdd_reg_lm and class lm, with specific print and plot methods

References

TODO

Examples

```
## Step 0: prepare data
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)

## Estimate a local probit:
house_rdd$y <- with(house_rdd, ifelse(y<quantile(y, 0.25), 0,1))
reg_bin_glm <- rdd_gen_reg(rdd_object=house_rdd, fun= glm, family=binomial(link='probit'))
print(reg_bin_glm)
summary(reg_bin_glm)
```

---

rdd_pred *RDD coefficient prediction*

Description

Function to predict the RDD coefficient in presence of covariate (without covariates, returns the same than rdd_coef)

Usage

```
rdd_pred(
  object,
  covdata,
  se.fit = TRUE,
  vcov. = NULL,
  newdata,
  stat = c("identity", "sum", "mean"),
  weights
)
```

Arguments

object	A RDD regression object
covdata	New data.frame specifying the values of the covariates, can have multiple rows.
se.fit	A switch indicating if standard errors are required.
vcov.	Specific covariance function (see package sandwich), by default uses the vcov
newdata	Another data on which to evaluate the x/D variables. Useful in very few cases.
stat	The statistic to use if there are multiple predictions, 'identity' just returns the single values, 'mean' averages them
weights	Eventual weights for the averaging of the predicted values.

Details

The function rdd_pred does a simple prediction of the RDD effect

$$RDDeffect = \mu(x, z, D = 1) - \mu(x, z, D = 0)$$

When there are no covariates (and z is irrelevant in the equation above), this amounts exactly to the usual RDD coefficient, shown in the outputs, or obtained with rdd_coef. If there were covariates, and if these covariates were estimated using the "include" *strategy* and with different coefficients left and right to the cutoff (i.e. had argument *slope* = "separate"), than the RDD effect is also dependent on the value of the covariate(s). rdd_pred allows to set the value of the covariate(s) at which to evaluate the RDD effect, by providing a data.frame with the values for the covariates. Note that the effect can be evaluated at multiple points, if you provide multiple rows of covdata.

In pressence of covariate-specific RDD effect, one may wish to estimate an average effect. This can be done by setting the argument stat='mean'. Weights can additionally be added, with the argument weights, to obtain a weighted-average of the predictions. Note however that in most cases, this will be equivalent to provide covariates at their (weighted) mean value, which will be much faster also!

Standard errors, obtained setting the argument se.fit=TRUE, are computed using following formula:

$$x_i \Omega x_i^{'}$$

where Ω is the estimated variance-covariance matrix (by default $\sigma^2(X^{'}X)^{-1}$ using vcov) and x_i is the input data (a mix of covdata and input data). If one wishes individual predictions, standard errors are simply obtained as the square of that diagonal matrix, whereas for mean/sum, covariances are taken into account.

Value

Returns the predicted value(s), and, if se.fit=TRUE, their standard errors.

References

Froehlich (2007) Regression discontinuity design with covariates, IZA discussion paper 3024

Examples

```
# Load data, add (artificial) covariates:
data(house)
n_Lee <- nrow(house)
z1 <- runif(n_Lee)
house_rdd <- rdd_data(y=y, x=x, data=house, covar=z1, cutpoint=0)

# estimation without covariates: rdd_pred is the same than rdd_coef:
reg_para <- rdd_reg_lm(rdd_object=house_rdd)

rdd_pred(reg_para)
rdd_coef(reg_para, allInfo=TRUE)

# estimation with covariates:
reg_para_cov <- rdd_reg_lm(rdd_object=house_rdd,
                           covariates='z1',
                           covar.opt=list(slope='separate') )

# should obtain same result as with RDestimate
rdd_pred(reg_para_cov, covdata=data.frame(z1=0))

# evaluate at mean of z1 (as comes from uniform)
rdd_pred(reg_para_cov, covdata=data.frame(z1=0.5))
```

`rdd_reg_lm` *Parametric polynomial estimator of the regression discontinuity*

Description

Compute a parametric polynomial regression of the ATE, possibly on the range specified by bandwidth

Usage

```
rdd_reg_lm(
  rdd_object,
  covariates = NULL,
  order = 1,
  bw = NULL,
  slope = c("separate", "same"),
 covar.opt = list(strategy = c("include", "residual"), slope = c("same", "separate"), bw
    = NULL),
  covar.strat = c("include", "residual"),
  weights
)
```

Arguments

rdd_object	Object of class rdd_data created by rdd_data
covariates	Formula to include covariates
order	Order of the polynomial regression.
bw	A bandwidth to specify the subset on which the parametric regression is estimated
slope	Whether slopes should be different on left or right (separate), or the same.
covar.opt	Options for the inclusion of covariates. Way to include covariates, either in the main regression (include) or as regressors of y in a first step (residual).
covar.strat	DEPRECATED, use covar.opt instead.
weights	Optional weights to pass to the lm function. Note this cannot be entered together with bw

Details

This function estimates the standard *discontinuity regression*:

$$Y = \alpha + \tau D + \beta_1(X - c) + \beta_2 D(X - c) + \epsilon$$

with τ the main parameter of interest. Several versions of the regression can be estimated, either restricting the slopes to be the same, i.e $\beta_1 = \beta_2$ (argument slope). The order of the polynomial in $X - c$ can also be adjusted with argument order. Note that a value of zero can be used, which corresponds to the simple *difference in means*, that one would use if the samples were random. Covariates can also be added in the regression, according to the two strategies discussed in Lee and Lemieux (2010, sec 4.5), through argument covar.strat:

include Covariates are simply added as supplementary regressors in the RD equation

residual The dependent variable is first regressed on the covariates only, then the RDD equation is applied on the residuals from this first step

The regression can also be estimated in a neighborhood of the cutpoint with the argument bw. This make the parametric regression resemble the non-parametric local kernel rdd_reg_np. Similarly, weights can also be provided (but not simultaneously to bw).

The returned object is a classical lm object, augmented with a RDDslot, so usual methods can be applied. As is done in general in R, heteroskeadsticity-robust inference can be done later on with the usual function from package **sandwich**. For the case of clustered observations a specific function clusterInf is provided.

Value

An object of class rdd_reg_lm and class lm, with specific print and plot methods

Examples

```
## Step 0: prepare data
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
```

```
## Step 2: regression
# Simple polynomial of order 1:
reg_para <- rdd_reg_lm(rdd_object=house_rdd)
print(reg_para)
plot(reg_para)

# Simple polynomial of order 4:
reg_para4 <- rdd_reg_lm(rdd_object=house_rdd, order=4)
reg_para4
plot(reg_para4)

# Restrict sample to bandwidth area:
bw_ik <- rdd_bw_ik(house_rdd)
reg_para_ik <- rdd_reg_lm(rdd_object=house_rdd, bw=bw_ik, order=4)
reg_para_ik
plot(reg_para_ik)
```

`rdd_reg_np` *Parametric polynomial estimator of the regression discontinuity*

Description

Compute a parametric polynomial regression of the ATE, possibly on the range specified by bandwidth

Usage

```
rdd_reg_np(
  rdd_object,
  covariates = NULL,
  bw = rdd_bw_ik(rdd_object),
  slope = c("separate", "same"),
  inference = c("np", "lm"),
  covar.opt = list(slope = c("same", "separate"), bw = NULL)
)
```

Arguments

`rdd_object`	Object of class rdd_data created by `rdd_data`
`covariates`	TODO
`bw`	A bandwidth to specify the subset on which the parametric regression is estimated
`slope`	Whether slopes should be different on left or right (separate), or the same.
`inference`	Type of inference to conduct: non-parametric one (`np`) or standard (`lm`). See details.
`covar.opt`	Options for the inclusion of covariates. Way to include covariates, either in the main regression (`include`) or as regressors of y in a first step (`residual`).

Value

An object of class rdd_reg_np and class lm, with specific print and plot methods

References

TODO

See Also

rdd_bw_ik Bandwidth selection using the plug-in bandwidth of Imbens and Kalyanaraman (2012)

Examples

```
## Step 0: prepare data
data(house)
house_rdd <- rdd_data(y=house$y, x=house$x, cutpoint=0)
## Step 2: regression
# Simple polynomial of order 1:
reg_nonpara <- rdd_reg_np(rdd_object=house_rdd)
print(reg_nonpara)
plot(reg_nonpara)
```

---

rot_bw *Bandwidth selector*

Description

implements dpill

Usage

```
rot_bw(object)
```

Arguments

object	object of class rdd_data

References

McCrary, Justin. (2008) 'Manipulation of the running variable in the regression discontinuity design: A density test,' *Journal of Econometrics*. 142(2): 698-714.

Examples

```
#No discontinuity
```

STAR_MHE *Transformation of the STAR dataset as used in Angrist and Pischke (2008)*

Description

Transformation of the STAR dataset as used in Table 8.2.1 of Angrist and Pischke (2008)

Format

A data frame containing 5743 observations and 6 variables. The first variable is from the original dataset, all other are created by Angrist and Pischke STAT code.

schidkn School ID in kindergarden (original variable, schoolidk in STAR)

pscore The propensity score (computed by A & P)

classid The id of the class (computed by A & P)

cs Class size (computed by A & P)

female, nwhite Various covariates (computed by A & P)

Details

). This is a transformation of the dataset from the project STAR (Student/Teacher Achievement Ratio. The full dataset is described and available in package AER, STAR. The transformed data was obtained using the STATA script krueger.do, obtained from Joshua Angrist website, on the webstar.dta.

Source

Data obtained using the script krueger.do on data webstar.rda, found on J. Angrist website

References

Krueger, A. (1999) 'Experimental Estimates Of Education Production Functions,' *The Quarterly Journal of Economics*, Vol. 114(2), pages 497-532, May.

Angrist, A. ad Pischke J-S (2008) *Mostly Harmless Econometrics: An Empiricist's Companion*, Princeton University press

See Also

STAR for the original dataset.

Examples

```
data(STAR_MHE)

# Compute the group means:
STAR_MHE_means <- aggregate(STAR_MHE[, c('classid', 'pscore', 'cs')],
                            by=list(STAR_MHE$classid), mean)

# Regression of means, with weighted average:
reg_krug_gls <- lm(pscore~cs, data=STAR_MHE_means, weights=cs)
coef(summary(reg_krug_gls))[2,2]
```

`vcovCluster` *Cluster Heteroskedasticity-consistent estimation of the covariance matrix.*

Description

Offer a cluster variant of the usual Heteroskedasticity-consistent

Usage

```
vcovCluster(object, clusterVar)

vcovCluster2(object, clusterVar1, clusterVar2)
```

Arguments

`object`	Object of class lm, from which rdd_reg also inherits.
`clusterVar`	The variable containing the cluster attributions.
`clusterVar1, clusterVar2`	The two cluster variables for the 2-cluster case.

Value

A matrix containing the covariance matrix estimate.

Author(s)

Mahmood Arai,

References

Cameron, C., Gelbach, J. and Miller, D. (2011) Robust Inference With Multiway Clustering, *Journal of Business and Economic Statistics*, vol. 29(2), pages 238-249. #' @references Wooldridge (2003) Cluster-sample methods in applied econometrics. *American Economic Review*, 93, p. 133-138

Arai, M. (2011) Cluster-robust standard errors using R, Note available https://web.archive.org/web/20230101000000/https://www.ne.su.se/polopoly_fs/1.216115.1426234213!/menu/standard/file/clustering1.pdf.

See Also

clusterInf for a direct function, allowing also alternative cluster inference methods.

Examples

```
data(STAR_MHE)
if(all(c(require(sandwich), require(lmtest)))){

# Run simple regression:
reg_krug <- lm(pscore~cs, data=STAR_MHE)

# Row 1 of Table 8.2.1, inference with standard vcovHC:
coeftest(reg_krug,vcov.=vcovHC(reg_krug, 'HC1'))[2,2]

# Row 4 of Table 8.2.1, inference with cluster vcovHC:
coeftest(reg_krug,vcov.=vcovCluster(reg_krug, clusterVar=STAR_MHE$classid))[2,2]
}
```

---

waldci *Confint allowing vcov*

Description

Version of vcov allowing for confint

Usage

```
waldci(x, parm = NULL, level = 0.95, vcov. = NULL, df = NULL, ...)
```

Arguments

x	Object of class lm or else
parm	specification of which parameters are to be given confidence intervals, see confint
level	the confidence level required, see confint()
vcov.	Specific covariance function to pass to coeftest. See help of sandwich
df	Degrees of freedom
...	Further arguments

Index

as.lm, 3, *4*
as.npreg, *3*
as.npreg *(*as.npregbw*)*, 4
as.npregbw, 4

clusterInf, 5, *29*, *34*
computePlacebo *(*plotPlacebo*)*, 15
covarTest_dis, 6, *8*
covarTest_mean, *7*, 7

dens_test, 9
dpill, *22*

gen_mc_ik, 10

house, 11

indh, 12

ks.test, *6*

npreg, *4*
npregbw, *4*

p.adjust, *6*, *8*
par, *13, 14*
plot, *13*
plot.rdd_data, 12
plotBin, 14
plotPlacebo, 15
plotPlaceboDens *(*plotPlacebo*)*, 15
plotSensi, 17

rdbwselect, *19*
rdd_bw_cct_estim, 19
rdd_bw_cct_plot, 20
rdd_bw_ik, *19, 20*, 21, *22*, *31*
rdd_bw_rsw, *13*, *21*, 22
rdd_coef, 23, *26, 27*
rdd_data, *19–22*, 23, *25*, *29, 30*
rdd_gen_reg, 25
rdd_pred, 26
rdd_reg_lm, *4*, *18*, 28
rdd_reg_np, *4*, *18*, *29*, 30
rddtools *(*rddtools-package*)*, 2
rddtools-package, 2
rdplot, *20*
rot_bw, 31

sandwich, *16*, *18*
STAR, *32*
STAR_MHE, 32

t.test, *8*

vcov, *27*
vcovCluster, *5*, 33
vcovCluster2 *(*vcovCluster*)*, 33

waldci, 34