

Package ‘npDoseResponse’

July 22, 2025

Type Package

Title Nonparametric Estimation and Inference on Dose-Response Curves

Version 0.1

Description

A novel integral estimator for estimating the causal effects with continuous treatments (or dose-response curves) and a localized derivative estimator for estimating the derivative effects. The inference on the dose-response curve and its derivative is conducted via nonparametric bootstrap. The reference paper is Zhang, Chen, and Giessing (2024) <[doi:10.48550/arXiv.2405.09003](https://doi.org/10.48550/arXiv.2405.09003)>.

URL <https://github.com/zhangyk8/npDoseResponse/>

BugReports <https://github.com/zhangyk8/npDoseResponse/issues>

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports parallel, stats

Suggests locpol

Maintainer Yikun Zhang <yikunzhang@foxmail.com>

NeedsCompilation no

Author Yikun Zhang [aut, cre] (ORCID: <<https://orcid.org/0000-0003-3905-6346>>),
Yen-Chi Chen [aut] (ORCID: <<https://orcid.org/0000-0002-4485-306X>>)

Repository CRAN

Date/Publication 2024-05-28 11:30:02 UTC

Contents

DerivEffect	2
DerivEffectBoot	4
IntegEst	6
IntegEstBoot	9
KernelRetrieval	11
LocalPolyReg	12

LocalPolyRegMain	14
RegAdjust	15
RoTBWLocalPoly	17

Index	19
--------------	-----------

DerivEffect	<i>The proposed localized derivative estimator.</i>
-------------	---

Description

This function implements our proposed estimator for estimating the derivative of a dose-response curve via Nadaraya-Watson conditional CDF estimator.

Usage

```
DerivEffect(
  Y,
  X,
  t_eval = NULL,
  h_bar = NULL,
  kernT_bar = "gaussian",
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  degree = 2,
  deriv_ord = 1,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  parallel = TRUE,
  cores = 6
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
t_eval	The m-dimensional vector for evaluating the derivative. (Default: t_eval = NULL. Then, t_eval = X[, 1], which consists of the observed treatment variables.)
h_bar	The bandwidth parameter for the Nadaraya-Watson conditional CDF estimator. (Default: h_bar = NULL. Then, the Silverman's rule of thumb is applied. See Chen et al. (2016) for details.)

kernT_bar	The name of the kernel function for the Nadaraya-Watson conditional CDF estimator. (Default: "gaussian".)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables in the local polynomial regression. (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1. It shouldn't be changed in most cases.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)
parallel	The indicator of whether the function should be parallel executed. (Default: parallel = TRUE.)
cores	The number of cores for parallel execution. (Default: cores = 6.)

Value

The estimated derivative of the dose-response curve evaluated at points `t_eval`.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Hall, P., Wolff, R. C., and Yao, Q. (1999) *Methods for Estimating A Conditional Distribution Function*. *Journal of the American Statistical Association*, 94 (445): 154-163.

Examples

```
library(parallel)
set.seed(123)
n <- 300

S2 <- cbind(2 * runif(n) - 1, 2 * runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
X2 <- cbind(T2, S2)

t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
```

```

if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}
theta_est2 = DerivEffect(Y2, X2, t_eval = t_qry2, h_bar = NULL,
                        kernT_bar = "gaussian", h = NULL, b = NULL,
                        C_h = 7, C_b = 3, print_bw = FALSE,
                        degree = 2, deriv_ord = 1, kernT = "epanechnikov",
                        kernS = "epanechnikov", parallel = TRUE, cores = num_workers)
plot(t_qry2, theta_est2, type="l", col = "blue", xlab = "t", lwd=5,
     ylab="(Estimated) derivative effects")
lines(t_qry2, 2*t_qry2 + 1, col = "red", lwd=3)
legend(-2, 5, legend=c("Estimated derivative", "True derivative"),
      fill = c("blue","red"))

```

DerivEffectBoot

Nonparametric bootstrap inference on the derivative effect via our localized derivative estimator.

Description

This function implements the nonparametric bootstrap inference on the derivative of a dose-response curve via our localized derivative estimator.

Usage

```

DerivEffectBoot(
  Y,
  X,
  t_eval = NULL,
  boot_num = 500,
  alpha = 0.95,
  h_bar = NULL,
  kernT_bar = "gaussian",
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  degree = 2,
  deriv_ord = 1,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  parallel = TRUE,

```

```
cores = 6
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
t_eval	The m-dimensional vector for evaluating the derivative. (Default: t_eval = NULL. Then, t_eval = X[, 1], which consists of the observed treatment variables.)
boot_num	The number of bootstrapping times. (Default: boot_num = 500.)
alpha	The confidence level of both the uniform confidence band and pointwise confidence interval. (Default: alpha = 0.95.)
h_bar	The bandwidth parameter for the Nadaraya-Watson conditional CDF estimator. (Default: h_bar = NULL. Then, the Silverman's rule of thumb is applied. See Chen et al. (2016) for details.)
kernT_bar	The name of the kernel function for the Nadaraya-Watson conditional CDF estimator. (Default: kernT_bar = "gaussian".)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables in the local polynomial regression. (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1. It shouldn't be changed in most cases.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)
parallel	The indicator of whether the function should be parallel executed. (Default: parallel = TRUE.)
cores	The number of cores for parallel execution. (Default: cores = 6.)

Value

A list that contains four elements.

theta_est	The estimated derivative of the dose-response curve evaluated at points t_eval.
theta_est_boot	The estimated derivative of the dose-response curve evaluated at points t_eval for all the bootstrap samples.
theta_alpha	The width of the uniform confidence band.
theta_alpha_var	The widths of the pointwise confidence bands at evaluation points t_eval.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Examples

```
set.seed(123)
n <- 300

S2 <- cbind(2 * runif(n) - 1, 2 * runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
X2 <- cbind(T2, S2)

t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}

# Increase bootstrap times "boot_num" to a larger integer in practice
theta_boot2 = DerivEffectBoot(Y2, X2, t_eval = t_qry2, boot_num = 3, alpha = 0.95,
                              h_bar = NULL, kernT_bar = "gaussian", h = NULL,
                              b = NULL, C_h = 7, C_b = 3, print_bw = FALSE,
                              degree = 2, deriv_ord = 1, kernT = "epanechnikov",
                              kernS = "epanechnikov", parallel = TRUE,
                              cores = num_workers)
```

IntegEst

The proposed integral estimator.

Description

This function implements our proposed integral estimator for estimating the dose-response curve.

Usage

```

IntegEst(
  Y,
  X,
  t_eval = NULL,
  h_bar = NULL,
  kernT_bar = "gaussian",
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  degree = 2,
  deriv_ord = 1,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  parallel = TRUE,
  cores = 6
)

```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
t_eval	The m-dimensional vector for evaluating the dose-response curve. (Default: t_eval = NULL. Then, t_eval = X[, 1], which consists of the observed treatment variables.)
h_bar	The bandwidth parameter for the Nadaraya-Watson conditional CDF estimator. (Default: h_bar = NULL. Then, the Silverman's rule of thumb is applied. See Chen et al. (2016) for details.)
kernT_bar	The name of the kernel function for the Nadaraya-Watson conditional CDF estimator. (Default: "gaussian".)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables in the local polynomial regression. (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1. It shouldn't be changed in most cases.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)

`parallel` The indicator of whether the function should be parallel executed. (Default: `parallel = TRUE`.)

`cores` The number of cores for parallel execution. (Default: `cores = 6`.)

Value

The estimated dose-response curve evaluated at points `t_eval`.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Examples

```
set.seed(123)
n <- 300

S2 <- cbind(2*runif(n) - 1, 2*runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
X2 <- cbind(T2, S2)

t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}
m_est2 = IntegEst(Y2, X2, t_eval = t_qry2, h_bar = NULL, kernT_bar = "gaussian",
  h = NULL, b = NULL, C_h = 7, C_b = 3, print_bw = FALSE,
  degree = 2, deriv_ord = 1, kernT = "epanechnikov",
  kernS = "epanechnikov", parallel = TRUE, cores = num_workers)

plot(t_qry2, m_est2, type="l", col = "blue", xlab = "t", lwd=5,
  ylab="(Estimated) dose-response curves")
lines(t_qry2, t_qry2^2 + t_qry2, col = "red", lwd=3)
legend(-2, 6, legend=c("Estimated curve", "True curve"), fill = c("blue","red"))
```

IntegEstBoot	<i>Nonparametric bootstrap inference on the dose-response curve via our integral estimator.</i>
--------------	---

Description

This function implements the nonparametric bootstrap inference on the dose-response curve via our integral estimator.

Usage

```
IntegEstBoot(
  Y,
  X,
  t_eval = NULL,
  boot_num = 500,
  alpha = 0.95,
  h_bar = NULL,
  kernT_bar = "gaussian",
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  degree = 2,
  deriv_ord = 1,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  parallel = TRUE,
  cores = 4
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
t_eval	The m-dimensional vector for evaluating the dose-response curve (Default: t_eval = NULL. Then, t_eval = X[, 1], which consists of the observed treatment variables.)
boot_num	The number of bootstrapping times. (Default: boot_num = 500.)
alpha	The confidence level of both the uniform confidence band and pointwise confidence interval. (Default: alpha = 0.95.)
h_bar	The bandwidth parameter for the Nadaraya-Watson conditional CDF estimator. (Default: h_bar = NULL. Then, the Silverman's rule of thumb is applied. See Chen et al. (2016) for details.)

kernT_bar	The name of the kernel function for the Nadaraya-Watson conditional CDF estimator. (Default: kernT_bar = "gaussian".)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables in the local polynomial regression. (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1. It shouldn't be changed in most cases.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)
parallel	The indicator of whether the function should be parallel executed. (Default: parallel = TRUE.)
cores	The number of cores for parallel execution. (Default: cores = 6.)

Value

A list that contains four elements.

m_est	The estimated dose-response curve evaluated at points t_eval.
m_est_boot	The estimated dose-response curve evaluated at points t_eval for all the bootstrap samples.
m_alpha	The width of the uniform confidence band.
m_alpha_var	The widths of the pointwise confidence bands at evaluation points t_eval.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Examples

```
set.seed(123)
n <- 300

S2 <- cbind(2 * runif(n) - 1, 2 * runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
```

```

X2 <- cbind(T2, S2)

t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}

# Increase bootstrap times "boot_num" to a larger integer in practice
m_boot2 = IntegEstBoot(Y2, X2, t_eval = t_qry2, boot_num = 3, alpha=0.95,
  h_bar = NULL, kernT_bar = "gaussian", h = NULL, b = NULL,
  C_h = 7, C_b = 3, print_bw = FALSE, degree = 2,
  deriv_ord = 1, kernT = "epanechnikov", kernS = "epanechnikov",
  parallel = TRUE, cores = num_workers)

```

KernelRetrieval

The helper function for retrieving a kernel function and its associated statistics.

Description

This function helps retrieve the commonly used kernel function, its second moment, and its variance based on the name.

Usage

```
KernelRetrieval(name)
```

Arguments

name	The lower-case full name of the kernel function.
------	--

Value

A list that contains three elements.

KernFunc	The interested kernel function.
sigmaK_sq	The second moment of the kernel function.
K_sq	The variance of the kernel function.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

Examples

```
kernel_result <- KernelRetrieval("epanechnikov")
kernT <- kernel_result$KernFunc
sigmaK_sq <- kernel_result$sigmaK_sq
K_sq <- kernel_result$K_sq
```

LocalPolyReg

*The (partial) local polynomial regression.***Description**

This function implements the (partial) local polynomial regression for estimating the conditional mean outcome function and its partial derivatives. We use higher-order local monomials for the treatment variable and first-order local monomials for the confounding variables.

Usage

```
LocalPolyReg(
  Y,
  X,
  x_eval = NULL,
  degree = 2,
  deriv_ord = 1,
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  kernT = "epanechnikov",
  kernS = "epanechnikov"
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
x_eval	The n*(d+1) matrix for evaluating the local polynomial regression estimates. (Default: x_eval = NULL. Then, x_eval = X.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1.)

h	The bandwidth parameter for the treatment/exposure variable. (Default: h = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h.)
b	The bandwidth vector for the confounding variables. (Default: b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_b.)
C_h	The scaling factor for the rule-of-thumb bandwidth parameter h.
C_b	The scaling factor for the rule-of-thumb bandwidth vector b.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)

Value

The estimated conditional mean outcome function or its partial derivatives evaluated at points `x_eval`.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Fan, J. and Gijbels, I. (1996) *Local Polynomial Modelling and its Applications*. Chapman & Hall/CRC.

Examples

```
library(parallel)
set.seed(123)
n <- 300
S2 <- cbind(2 * runif(n) - 1, 2 * runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
X2 <- cbind(T2, S2)
t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}
```

```
Y_est2 = LocalPolyReg(Y2, X2, x_eval = NULL, degree = 2, deriv_ord = 0,
                      h = NULL, b = NULL, C_h = 7, C_b = 3, print_bw = TRUE,
                      kernT = "epanechnikov", kernS = "epanechnikov")
```

LocalPolyRegMain

The main function of the (partial) local polynomial regression.

Description

This function implements the main part of the (partial) local polynomial regression for estimating the conditional mean outcome function and its partial derivatives.

Usage

```
LocalPolyRegMain(
  Y,
  X,
  x_eval = NULL,
  degree = 2,
  deriv_ord = 1,
  h = NULL,
  b = NULL,
  kernT = "epanechnikov",
  kernS = "epanechnikov"
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
x_eval	The n*(d+1) matrix for evaluating the local polynomial regression estimates. (Default: x_eval = NULL. Then, x_eval = X.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1.)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)

Value

The estimated conditional mean outcome function or its partial derivatives evaluated at points `x_eval`.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Fan, J. and Gijbels, I. (1996) *Local Polynomial Modelling and its Applications*. Chapman & Hall/CRC.

RegAdjust

The regression adjustment estimator of the dose-response curve.

Description

This function implements the standard regression adjustment or G-computation estimator of the dose-response curve or its derivative via (partial) local polynomial regression.

Usage

```
RegAdjust(
  Y,
  X,
  t_eval = NULL,
  h = NULL,
  b = NULL,
  C_h = 7,
  C_b = 3,
  print_bw = TRUE,
  degree = 2,
  deriv_ord = 0,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  parallel = TRUE,
  cores = 6
)
```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
t_eval	The m-dimensional vector for evaluating the dose-response curve. (Default: t_eval = NULL. Then, t_eval = X[, 1], which consists of the observed treatment variables.)
h, b	The bandwidth parameters for the treatment/exposure variable and confounding variables (Default: h = NULL, b = NULL. Then, the rule-of-thumb bandwidth selector in Eq. (A1) of Yang and Tschernig (1999) is used with additional scaling factors C_h and C_b, respectively.)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.
print_bw	The indicator of whether the current bandwidth parameters should be printed to the console. (Default: print_bw = TRUE.)
degree	Degree of local polynomials. (Default: degree = 2.)
deriv_ord	The order of the estimated derivative of the conditional mean outcome function. (Default: deriv_ord = 1.)
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)
parallel	The indicator of whether the function should be parallel executed. (Default: parallel = TRUE.)
cores	The number of cores for parallel execution. (Default: cores = 6.)

Value

The estimated dose-response curves or its derivatives evaluated at points t_eval.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Fan, J. and Gijbels, I. (1996) *Local Polynomial Modelling and its Applications*. Chapman & Hall/CRC.

Examples

```
library(parallel)
set.seed(123)
n <- 300

S2 <- cbind(2 * runif(n) - 1, 2 * runif(n) - 1)
Z2 <- 4 * S2[, 1] + S2[, 2]
```



```

E2 <- 0.2 * runif(n) - 0.1
T2 <- cos(pi * Z2^3) + Z2 / 4 + E2
Y2 <- T2^2 + T2 + 10 * Z2 + rnorm(n, mean = 0, sd = 1)
X2 <- cbind(T2, S2)

t_qry2 = seq(min(T2) + 0.01, max(T2) - 0.01, length.out = 100)
chk <- Sys.getenv("_R_CHECK_LIMIT_CORES_", "")
if (nzchar(chk) && chk == "TRUE") {
  # use 2 cores in CRAN/Travis/AppVeyor
  num_workers <- 2L
} else {
  # use all cores in devtools::test()
  num_workers <- parallel::detectCores()
}
Y_RA2 = RegAdjust(Y2, X2, t_eval = t_qry2, h = NULL, b = NULL, C_h = 7, C_b = 3,
  print_bw = FALSE, degree = 2, deriv_ord = 0,
  kernT = "epanechnikov", kernS = "epanechnikov",
  parallel = TRUE, cores = num_workers)

```

RoTBWLocalPoly

The rule-of-thumb bandwidth selector for the (partial) local polynomial regression.

Description

This function implements the rule-of-thumb bandwidth selector for the (partial) local polynomial regression.

Usage

```

RoTBWLocalPoly(
  Y,
  X,
  kernT = "epanechnikov",
  kernS = "epanechnikov",
  C_h = 7,
  C_b = 3
)

```

Arguments

Y	The input n-dimensional outcome variable vector.
X	The input n*(d+1) matrix. The first column of X stores the treatment/exposure variables, while the other d columns are confounding variables.
kernT, kernS	The names of kernel functions for the treatment/exposure variable and confounding variables. (Default: kernT = "epanechnikov", kernS = "epanechnikov".)
C_h, C_b	The scaling factors for the rule-of-thumb bandwidth parameters.

Value

A list that contains two elements.

h The rule-of-thumb bandwidth parameter for the treatment/exposure variable.

b The rule-of-thumb bandwidth vector for the confounding variables.

Author(s)

Yikun Zhang, <yikunzhang@foxmail.com>

References

Zhang, Y., Chen, Y.-C., and Giessing, A. (2024) *Nonparametric Inference on Dose-Response Curves Without the Positivity Condition*. <https://arxiv.org/abs/2405.09003>.

Yang, L. and Tschernig, R. (1999). *Multivariate Bandwidth Selection for Local Linear Regression*. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 61(4), 793-815.

Index

- * **(partial)**
 - LocalPolyReg, [12](#)
 - LocalPolyRegMain, [14](#)
 - RegAdjust, [15](#)
- * **and**
 - KernelRetrieval, [11](#)
- * **associated**
 - KernelRetrieval, [11](#)
- * **a**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
- * **bandwidth**
 - RoTBWLocalPoly, [17](#)
- * **bootstrap**
 - DerivEffectBoot, [4](#)
 - IntegEstBoot, [9](#)
- * **curve**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
 - IntegEst, [6](#)
 - IntegEstBoot, [9](#)
- * **derivative**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
- * **dose-response**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
 - IntegEst, [6](#)
 - IntegEstBoot, [9](#)
- * **estimator**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
 - IntegEst, [6](#)
 - IntegEstBoot, [9](#)
- * **for**
 - RoTBWLocalPoly, [17](#)
- * **function**
 - KernelRetrieval, [11](#)
 - LocalPolyRegMain, [14](#)
- * **inference**
 - DerivEffectBoot, [4](#)
 - IntegEstBoot, [9](#)
- * **integral**
 - IntegEst, [6](#)
 - IntegEstBoot, [9](#)
- * **its**
 - KernelRetrieval, [11](#)
- * **kernel**
 - KernelRetrieval, [11](#)
- * **localized**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
- * **local**
 - LocalPolyReg, [12](#)
 - LocalPolyRegMain, [14](#)
 - RegAdjust, [15](#)
 - RoTBWLocalPoly, [17](#)
- * **main**
 - LocalPolyRegMain, [14](#)
- * **of**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
 - IntegEst, [6](#)
 - LocalPolyRegMain, [14](#)
- * **on**
 - DerivEffectBoot, [4](#)
 - IntegEstBoot, [9](#)
- * **our**
 - DerivEffectBoot, [4](#)
 - IntegEstBoot, [9](#)
- * **polynomial**
 - LocalPolyReg, [12](#)
 - LocalPolyRegMain, [14](#)
 - RegAdjust, [15](#)
 - RoTBWLocalPoly, [17](#)
- * **regression**
 - LocalPolyReg, [12](#)
 - LocalPolyRegMain, [14](#)

- RegAdjust, [15](#)
- RoTBWLocalPoly, [17](#)
- * **retrieve**
 - KernelRetrieval, [11](#)
- * **rule-of-thumb**
 - RoTBWLocalPoly, [17](#)
- * **selector**
 - RoTBWLocalPoly, [17](#)
- * **statistics**
 - KernelRetrieval, [11](#)
- * **the**
 - DerivEffect, [2](#)
 - DerivEffectBoot, [4](#)
 - IntegEst, [6](#)
 - IntegEstBoot, [9](#)
 - LocalPolyRegMain, [14](#)
- * **through**
 - DerivEffectBoot, [4](#)
 - IntegEstBoot, [9](#)

DerivEffect, [2](#)

DerivEffectBoot, [4](#)

IntegEst, [6](#)

IntegEstBoot, [9](#)

KernelRetrieval, [11](#)

LocalPolyReg, [12](#)

LocalPolyRegMain, [14](#)

RegAdjust, [15](#)

RoTBWLocalPoly, [17](#)