Package 'eiPartialID'

April 11, 2019

Type Package

Title Ecological Regression with Partial Identification

Version 0.1.2

Author Wenxin Jiang, Gary King, Allen Schmaltz, Martin A. Tanner

Maintainer Allen Schmaltz <schmaltz@fas.harvard.edu>

Contact
<wjiang@northwestern.edu,king@harvard.edu,schmaltz@fas.harvard.edu,mat132@northwestern.edu>

Description Estimate district-level bounds for 2x2 ecological inference based on the approach described in the forthcoming article Jiang et al. (2019), ``Ecological Regression with Partial Identification'', Political Analysis. Interval data regression is used to bound the nonidentified regression parameter in a linear contextual effects model, from which district-level bounds are derived. The approach here can be useful as a baseline of comparison for future work on ecological inference.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports sandwich, stats, MASS, eco

RoxygenNote 6.1.1.9000

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-04-11 16:32:38 UTC

R topics documented:

---

bounds *Compute and evaluate bounds according to Jiang et al. 2019, illustrating usage.*

Description

`bounds()` calculates district-level bounds across varying coverage probabilities, after applying the heuristics presented in Jiang et al. 2019. This is a simple wrapper around calling generateBounds() followed by evaluateBounds(). Here, the returned object only contains the CI_0.5 bounds.

Usage

```
bounds(x, t, n, trueBetaB = NULL)
```

Arguments

`x`	Numeric (double-precision) vector. Contains the proportion of variable X in each precinct (or analagous geographic unit)
`t`	Numeric (double-precision) vector. Contains the proportion of variable T in each precinct (or analagous geographic unit)
`n`	Numeric (double-precision) vector. Contains the number of elements (people/households/etc.) in each precinct (or analagous geographic unit)
`trueBetaB`	Numeric (double-precision) vector. Contains the true conditional values (beta_i) in each precinct (or analagous geographic unit). Optional. Default NULL.

Value

List object with the CI_0.5 bounds:

CI_0.5_lower CI_0.5 lower bound

CI_0.5_upper CI_0.5 upper bound

CI_0.5_isSelected If FALSE, proposed bound was not rejected by the heuristic (if TRUE, bounds are reverted to the DD bounds)

CI_0.5_widthRatio |CI_x|/|DD|

CI_0.5_nominalCoverage Nominal coverage (1-pnorm(-0.5))

Optional: CI_0.5_truthCaptured If true district Beta is provided as an argument to bounds(), then this variable contains a boolean for whether or not the true value was captured in the proposed CI_0.5.

Examples

```
library("MASS")
library("eco")
data("census")
inputDataSet <- census
x <- inputDataSet$X
t <- inputDataSet$Y
n <- inputDataSet$N
trueBetaB <- inputDataSet$W1
outputList <- bounds(x, t, n, trueBetaB=trueBetaB)
print(outputList)

# > print(outputList)
# $CI_0.5_lower
# [1] 0.5893336
#
# $CI_0.5_upper
# [1] 0.8262426
#
# $CI_0.5_isSelected
# [1] TRUE
#
# $CI_0.5_widthRatio
# [1] 0.5404046
#
# $CI_0.5_nominalCoverage
# [1] 0.6914625
#
# $CI_0.5_truthCaptured
# [1] TRUE
```

`calcSummaryOutputValues` *Internal/private method. Compute bounds and summary statistics according to Jiang et al. 2019*

Description

`calcSummaryOutputValues()` is an internal/private helper method for calculating the bounds. Called by generateBounds().

Usage

```
calcSummaryOutputValues(x, t, n, trueBetaB = NULL,
  useXRangeOffset = TRUE, returnAdditionalStats = FALSE)
```

Arguments

x	Numeric (double-precision) vector. Contains the proportion of variable X in each precinct (or analagous geographic unit)
t	Numeric (double-precision) vector. Contains the proportion of variable T in each precinct (or analagous geographic unit)
n	Numeric (double-precision) vector. Contains the number of elements (people/households/etc.) in each precinct (or analagous geographic unit)
trueBetaB	Numeric (double-precision) vector. Contains the true conditional values (beta_i) in each precinct (or analagous geographic unit). Optional. Default NULL.
useXRangeOffset	boolean If True, an offset of 0.00001 is applied to l and u to avoid division by 0 in subsequent calculations. Default TRUE
returnAdditionalStats	boolean If True, additional summary statistics are generated. Default FALSE.

Value

List object with the bounds and summary statistics

Examples

```
## Not run:
outputList <- calcSummaryOutputValues(x, t, n, NULL, TRUE, FALSE)

## End(Not run)
```

---

calcSummaryOutputValues_

Internal/private method. Compute bounds and summary statistics according to Jiang et al. 2019

Description

`calcSummaryOutputValues_()` is an internal/private helper method for calculating the bounds. Called by calcSummaryOutputValues().

Usage

```
calcSummaryOutputValues_(x, t, n, useXRangeOffset = TRUE)
```

Arguments

x	Numeric (double-precision) vector. Contains the proportion of variable X in each precinct (or analagous geographic unit)
t	Numeric (double-precision) vector. Contains the proportion of variable T in each precinct (or analagous geographic unit)
n	Numeric (double-precision) vector. Contains the number of elements (people/households/etc.) in each precinct (or analagous geographic unit)
useXRangeOffset	boolean If True, an offset of 0.00001 is applied to l and u to avoid division by 0 in subsequent calculations. Default TRUE.

Value

List object with the bounds and summary statistics

Examples

```
## Not run:
outputList <- calcSummaryOutputValues_(x, t, n, TRUE)

## End(Not run)
```

---

evaluateBounds *Evaluate computed bounds, across confidence levels, applying the selection heuristic of Jiang et al. 2019*

Description

`evaluateBounds()` calculates the bounds across confidence levels and generates the width-ratio relative to the deterministic DD bounds using the bounds generated by generateBounds(), after applying the selection heuristic of Jiang et al. 2019. If the true district B is provided, the capture of the true value is checked.

Usage

```
evaluateBounds(outputListFromGenerateBounds)
```

Arguments

outputListFromGenerateBounds	List returned by generateBounds()

Value

List object with the bounds indexed across confidence levels:

x_for_x_in_CI_x c(0.00,0.25,0.50,0.75, 1.00, 1.25, 1.50, 1.75, 2.00), which corresponds to CI_0, CI_0.25, ..., CI_2.00 (the following vectors are parallel in indexes)

CI_x_lower CI_x lower bound

CI_x_upper CI_x upper bound

CI_x_isSelected If FALSE, proposed bound was not rejected by the heuristic (if TRUE, bounds are reverted to the DD bounds)

CI_x_widthRatio |CI_x|/|DD|

CI_x_nominalCoverage Nominal coverage (1-pnorm(-x_for_x_in_CI_x))

Optional: CI_x_truthCaptured If true district Beta is provided in outputListFromGenerateBounds, then this vector contains a boolean for whether or not the true value was captured in the proposed CI_x.

Examples

```
library("MASS")
library("eco")
data("census")
inputDataSet <- census
x <- inputDataSet$X
t <- inputDataSet$Y
n <- inputDataSet$N
trueBetaB <- inputDataSet$W1
outputList <- generateBounds(x, t, n, trueBetaB=trueBetaB, useXRangeOffset=TRUE,
    returnAdditionalStats=FALSE, printSummary=TRUE)
summaryOutputList <- evaluateBounds(outputList)

# $x$ & Nominal coverage (\Phi(x)) & True B in CI_x & Width-ratio: |Proposed width|/|DD| &
#     Reverted to DD & Proposed Lower & Proposed Upper \\
# 0.00 & 0.5000 & TRUE & 0.4653 & FALSE & 0.6061 & 0.8101\\
# 0.25 & 0.5987 & TRUE & 0.5028 & FALSE & 0.5977 & 0.8182\\
# 0.50 & 0.6915 & TRUE & 0.5404 & FALSE & 0.5893 & 0.8262\\
# 0.75 & 0.7734 & TRUE & 0.5780 & FALSE & 0.5809 & 0.8343\\
# 1.00 & 0.8413 & TRUE & 0.6155 & FALSE & 0.5726 & 0.8424\\
# 1.25 & 0.8944 & TRUE & 0.6531 & FALSE & 0.5642 & 0.8505\\
# 1.50 & 0.9332 & TRUE & 0.6906 & FALSE & 0.5558 & 0.8586\\
# 1.75 & 0.9599 & TRUE & 0.7282 & FALSE & 0.5474 & 0.8666\\
# 2.00 & 0.9772 & TRUE & 0.7657 & FALSE & 0.5390 & 0.8747\\

# For example, CI_0.5 (0.5893336 0.8262426) corresponds to
#     c(summaryOutputList$CI_x_lower[3], summaryOutputList$CI_x_upper[3])
```

`generateBounds` *Compute bounds and summary statistics according to Jiang et al. 2019*

Description

`generateBounds()` calculates district-level bounds. The returned object can be passed to evaluateBounds() to generate bounds across varying coverage probabilities and to apply the heuristics presented in Jiang et al. 2019.

Usage

```
generateBounds(x, t, n, trueBetaB = NULL, useXRangeOffset = TRUE,
  returnAdditionalStats = FALSE, printSummary = TRUE)
```

Arguments

`x`	Numeric (double-precision) vector. Contains the proportion of variable X in each precinct (or analagous geographic unit)
`t`	Numeric (double-precision) vector. Contains the proportion of variable T in each precinct (or analagous geographic unit)
`n`	Numeric (double-precision) vector. Contains the number of elements (people/households/etc.) in each precinct (or analagous geographic unit)
`trueBetaB`	Numeric (double-precision) vector. Contains the true conditional values (beta_i) in each precinct (or analagous geographic unit). Optional. Default NULL.
`useXRangeOffset`	boolean If True, an offset of 0.00001 is applied to l and u to avoid division by 0 in subsequent calculations. Default TRUE
`returnAdditionalStats`	boolean If True, additional summary statistics are generated. Default FALSE.
`printSummary`	boolean If True, the DD bounds, l and u, CI_0, CI_1, width-ratio, and (optionally) true district B are output to standard out. Default TRUE.

Value

List object with the bounds and summary statistics:

nx1 Total elements (people/households/etc.) of variable X across all geographic units

hbdl0 CI_0 lower bound

hbdu0 CI_0 upper bound

cil CI_1 lower bound

cir CI_1 upper bound

bdl Duncan-Davis lower bound

bdu Duncan-Davis upper bound

Optional: bd True district Beta

Examples

```
library("MASS")
library("eco")
data("census")
inputDataSet <- census
x <- inputDataSet$X
t <- inputDataSet$Y
n <- inputDataSet$N
trueBetaB <- inputDataSet$W1
outputList <- generateBounds(x, t, n, trueBetaB=trueBetaB, useXRangeOffset=TRUE,
    returnAdditionalStats=FALSE, printSummary=TRUE)

# True B: 0.674809
# Duncan-Davis bounds: [0.535618, 0.974010]
# [l,u]=[min(X_i),max(X_i)]: [0.050810, 0.939290]
# CI_0=[Bl_hat, Bu_hat]: [0.606101, 0.810082]
# CI_1: [0.572566, 0.842403]
# Width-ratio: |CI_0|/|DD|: 0.465295
```

---

generateDataExample3 *Simulate data; Example 3 from Jiang et al. 2019.*

Description

generateDataExample3 generates simulated data for example simulation 3 in Jiang et al. 2019.

Usage

```
generateDataExample3()
```

Value

List object with the following attributes: x (proportion of X in each geographic unit); t (proportion of T in each geographic unit); n (population of each geographic unit); bd (the true district B);

Examples

```
generatedData <- generateDataExample3()
```

generateDataExample4 *Simulate data; Example 4 from Jiang et al. 2019.*

Description

generateDataExample4 generates simulated data for example simulation 4 in Jiang et al. 2019.

Usage

```
generateDataExample4()
```

Value

List object with the following attributes: x (proportion of X in each geographic unit); t (proportion of T in each geographic unit); n (population of each geographic unit); bd (the true district B);

Examples

```
generatedData <- generateDataExample4()
```

generateDataExample5 *Simulate data; Example 5 from Jiang et al. 2019.*

Description

generateDataExample5 generates simulated data for example simulation 5 in Jiang et al. 2019.

Usage

```
generateDataExample5()
```

Value

List object with the following attributes: x (proportion of X in each geographic unit); t (proportion of T in each geographic unit); n (population of each geographic unit); bd (the true district B);

Examples

```
generatedData <- generateDataExample5()
```

runExampleSimulations *Run illustrative simulations.*

Description

runExampleSimulations generates bounds for the example simulations in Jiang et al. 2019.

Usage

```
runExampleSimulations()
```

Value

No explicit return values. The summary of the simulations will be printed to standard out.

Examples

```
runExampleSimulations()
```

Index