Package ‘ivdoctr’

December 5, 2021

Title Ensures Mutually Consistent Beliefs When Using IVs

Version 1.0.1

Description Uses data and researcher's beliefs on measurement error and instrumental variable (IV) endogeneity to generate the space of consistent beliefs across measurement error, instrument endogeneity, and instrumental relevance for IV regressions.


License CC0

LazyData TRUE

Depends R (>= 2.10)

Imports AER, coda, data.table, graphics, MASS, Rcpp (>= 0.11.6), rgl, sandwich, stats

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, haven, MCMCpack, knitr, rmarkdown

RoxygenNote 7.1.2

Encoding UTF-8

NeedsCompilation yes

BugReports https://github.com/emallickhossain/ivdoctr/issues

VignetteBuilder knitr

Author Frank DiTraglia [aut],
Mallick Hossain [aut, cre]

Maintainer Mallick Hossain <emallickhossain@gmail.com>

Repository CRAN

Date/Publication 2021-12-05 16:00:02 UTC

R topics documented:

afghan ............................................................. 3
b_functionA3 ..................................................... 4
candidate1 ...................................................... 4
**afghan**

Burde and Linden (2013, AEJ Applied) Dataset

**Description**

Replicates IV using controls from Table 2

**Usage**

afghan

**Format**

A data frame with 687 rows and 17 variables:

- **enrolled** Indicator if child is enrolled in formal school. Outcome.
- **testscore** Normalized test score
- **buildschool** Indicator if village is treated. Instrument.
- **headchild** Indicator if child is child of head of household
- **nhh** Number of household members
- **female** Female indicator
- **age** Child’s age
- **yrsvill** Time family has lived in village
- **farsi** Indicator for speaking Farsi
- **tajik** Indicator for speaking Tajik
- **farmers** Indicator for if head of household is a farmer
- **land** Number of jeribs of land owned
- **agehead** Head of household age
- **educhead** Years of education for head of household
- **sheep** Number of sheep and goats owned
- **chagcharan** Indicator if village is in Chagcharan district
- **distschool** Distance to nearest non-community based school

**Source**

Provided by author.

**References**

**Description**

B function from Proposition A3

**Usage**

`b_functionA3(obs_draws, g, psi)`

**Arguments**

- `obs_draws`: Row of the data.frame of observable draws
- `g`: Value from g function
- `psi`: Psi value

**Value**

A min and a max of the B function

---

**Description**

Evaluates the corners given user bounds. Vectorized wrt multiple draws of obs.

**Usage**

`candidate1(r_TstarU_lower, r_TstarU_upper, k_lower, k_upper, obs)`

**Arguments**

- `r_TstarU_lower`: Vector of lower bounds of endogeneity
- `r_TstarU_upper`: Vector of upper bounds of endogeneity
- `k_lower`: Vector of lower bounds on measurement error
- `k_upper`: Vector of upper bounds on measurement error
- `obs`: Observables generated by get_observables

**Value**

List containing vector of lower bounds and vector of upper bounds of r_uz
candidate2

Evaluates the edge where k is on the boundary. Vectorized wrt multiple draws of obs.

Description
Evaluates the edge where k is on the boundary. Vectorized wrt multiple draws of obs.

Usage
candidate2(r_TstarU_lower, r_TstarU_upper, k_lower, k_upper, obs)

Arguments
- r_TstarU_lower: Vector of lower bounds of endogeneity
- r_TstarU_upper: Vector of upper bounds of endogeneity
- k_lower: Vector of lower bounds on measurement error
- k_upper: Vector of upper bounds on measurement error
- obs: Observables generated by get_observables

Value
List containing vector of lower bounds and vector of upper bounds of r_uz

candidate3

Evaluates the edge where r_TstarU is on the boundary.

Description
Evaluates the edge where r_TstarU is on the boundary.

Usage
candidate3(r_TstarU_lower, r_TstarU_upper, k_lower, k_upper, obs)

Arguments
- r_TstarU_lower: Vector of lower bounds of endogeneity
- r_TstarU_upper: Vector of upper bounds of endogeneity
- k_lower: Vector of lower bounds on measurement error
- k_upper: Vector of upper bounds on measurement error
- obs: Observables generated by get_observables

Value
List containing vector of lower bounds and vector of upper bounds of r_uz
### collapse_3d_array

**Description**

Collapse 3-d array to matrix

**Usage**

```r
collapse_3d_array(myarray)
```

**Arguments**

- `myarray` A three-dimensional array.

**Value**

Matrix with the 3rd dimension appended as rows to the matrix

### colonial

**Acemoglu, Johnson, and Robinson (2001) Dataset**

**Description**

Cross-country dataset used to construct Table 4 of Acemoglu, Johnson & Robinson (2001).

**Usage**

```r
colonial
```

**Format**

A data frame with 64 rows and 9 variables:

- `shortnam` three letter country abbreviation, e.g. AUS for Australia
- `africa` dummy variable =1 if country is in Africa
- `lat_abst` absolute distance to equator (scaled between 0 and 1)
- `rich4` dummy variable, =1 for "Neo-Europes" (AUS, CAN, NZL, USA)
- `avexpr` Average protection against expropriation risk. Measures risk of government appropriation of foreign private investment on a scale from 0 (least risk) to 10 (most risk). Averaged over all years from 1985-1995.
- `logpgp95` Natural logarithm of per capita GDP in 1995 at purchasing power parity
- `logem4` Natural logarithm of European settler mortality
- `asia` dummy variable, =1 if country is in Asia
- `loghjypl` Natural logarithm of output per worker in 1988
source

http://economics.mit.edu/faculty/acemoglu/data/ajr2001

references

https://www.aeaweb.org/articles.php?doi=10.1257/aer.91.5.1369

draw_bounds

Computes bounds for simulated data

Description

This function takes data and user restrictions on measurement error and endogeneity and simulates data and the resulting bounds on instrument validity.

Usage

draw_bounds( 
    y_name, 
    T_name, 
    z_name, 
    data, 
    controls = NULL, 
    r_TstarU_restriction = NULL, 
    k_restriction = NULL, 
    n_draws = 5000
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y_name</td>
<td>Character vector of the name of the dependent variable</td>
</tr>
<tr>
<td>T_name</td>
<td>Character vector of the names of the preferred regressors</td>
</tr>
<tr>
<td>z_name</td>
<td>Character vector of the names of the instrumental variables</td>
</tr>
<tr>
<td>data</td>
<td>Data to be analyzed</td>
</tr>
<tr>
<td>controls</td>
<td>Character vector containing the names of the exogenous regressors</td>
</tr>
<tr>
<td>r_TstarU_restriction</td>
<td>2 element vector of bounds on r_TstarU</td>
</tr>
<tr>
<td>k_restriction</td>
<td>2-element vector of bounds on kappa</td>
</tr>
<tr>
<td>n_draws</td>
<td>Integer number of simulations to draw</td>
</tr>
</tbody>
</table>

Value

List containing simulated data observables (covariances, correlations, and R-squares), indications of whether the identified set is empty, the unrestricted and restricted bounds on instrumental relevance, instrumental validity, and measurement error.
draw_observables  
*Simulates different data draws*

**Description**
This function takes the data and simulates potential draws of data from the properties of the observed data.

**Usage**
```r
draw_observables(y_name, T_name, z_name, data, controls = NULL, n_draws = 5000)
```

**Arguments**
- `y_name`: Character vector of the name of the dependent variable
- `T_name`: Character vector of the names of the preferred regressors
- `z_name`: Character vector of the names of the instrumental variables
- `data`: Data to be analyzed
- `controls`: Character vector containing the names of the exogenous regressors
- `n_draws`: Integer number of simulations to draw

**Value**
Data frame containing covariances, correlations, and R-squares for each data simulation

---

draw_sigma_jeffreys  
*Draws covariance matrix using the Jeffrey’s Prior*

**Description**
Draws covariance matrix using the Jeffrey’s Prior

**Usage**
```r
draw_sigma_jeffreys(y, Tobs, z, k, n_draws)
```

**Arguments**
- `y`: Vector of dependent variable
- `Tobs`: Matrix containing data for the preferred regressor
- `z`: Matrix containing data for the instrumental variable
- `k`: Number of covariates, including the intercept
- `n_draws`: Integer number of draws to perform
**Value**

Array of covariance matrix draws

---

**format_est**

*Creates LaTeX code for parameter estimates*

---

**Description**

 Creates LaTeX code for parameter estimates

**Usage**

`format_est(est)`

**Arguments**

`est` Number

**Value**

LaTeX string for the number

---

**format_HPDI**

*Creates LaTeX code for the HPDI*

---

**Description**

 Creates LaTeX code for the HPDI

**Usage**

`format_HPDI(bounds)`

**Arguments**

`bounds` 2-element vector of the upper and lower HPDI bounds

**Value**

LaTeX string of the HPDI
format_se

*Description*

Creates LaTeX code for the standard error

*Usage*

format_se(se)

*Arguments*

| se | Standard error |

*Value*

LaTeX string for the standard error

getCoverage

*Description*

Computes coverage of list of intervals

*Usage*

getCoverage(data, guess)

*Arguments*

| data | 2-column data frame of confidence intervals |
| guess | 2-element vector of confidence interval |

*Value*

Coverage percentage
**getInterval**

Generates smallest covering interval

**Description**
Generates smallest covering interval

**Usage**
```r
getInterval(data, center, conf = 0.9, tol = 1e-06)
```

**Arguments**
- `data`: 2-column data frame of confidence intervals
- `center`: 2-element vector to center coverage interval
- `conf`: Confidence level
- `tol`: Tolerance level for convergence

**Value**
2-element vector of confidence interval

---

**get_alpha_bounds**

Computes $a_0$ and $a_1$ bounds

**Description**
Computes $a_0$ and $a_1$ bounds

**Usage**
```r
get_alpha_bounds(draws, p)
```

**Arguments**
- `draws`: data.frame of observables of simulated data
- `p`: Treatment probability from binary data

**Value**
List of alpha bounds
**get_beta**

*Solves for beta*

**Description**

This function solves for beta given \( r_{TstarU} \) and \( \kappa \). It handles 3 potential cases when beta must be evaluated: 1. Across multiple simulations, but given the same \( r_{TstarU} \) and \( \kappa \). 2. For multiple simulations, each with a value of \( r_{TstarU} \) and \( \kappa \). 3. For one simulation across a grid of \( r_{TstarU} \) and \( \kappa \).

**Usage**

```r
get_beta(r_TstarU, k, obs)
```

**Arguments**

- `r_TstarU`: Vector of \( r_{TstarU} \) values
- `k`: Vector of kappa values
- `obs`: Observables generated by `get_observables`

**Value**

Vector of betas

---

**get_beta_bounds_binary**

*Returns beta bounds in binary case using grid search*

**Description**

Returns beta bounds in binary case using grid search

**Usage**

```r
get_beta_bounds_binary(obs_draws, p, r_TstarU_restriction)
```

**Arguments**

- `obs_draws`: Row of the data.frame of observable draws
- `p`: Treatment probability from data
- `r_TstarU_restriction`: 2-element vector of restrictions on \( r_{TstarU} \)

**Value**

Min and max values for beta
**get_beta_bounds_binary_post**

*Generates beta bounds off of beta draws*

**Description**

Generates beta bounds off of beta draws

**Usage**

```python
get_beta_bounds_binary_post(draws, n_observables)
```

**Arguments**

- **draws**: Posterior draws
- **n_observables**: Number of observable draws

**Value**

Upper and lower bounds of beta based on posterior draws

---

**get_bounds_unrest**

*Wrapper function combines all unrestricted bounds together. Vectorized*

**Description**

Wrapper function combines all unrestricted bounds together. Vectorized

**Usage**

```python
get_bounds_unrest(obs)
```

**Arguments**

- **obs**: Observables generated by get_observables

**Value**

List of unrestricted bounds for r_TstarU, r uz, and kappa
get_estimates
Computes OLS and IV estimates

Description
Computes OLS and IV estimates

Usage
get_estimates(y_name, T_name, z_name, data, controls = NULL, robust = FALSE)

Arguments
- y_name: Character vector of the name of the dependent variable
- T_name: Character vector of the names of the preferred regressors
- z_name: Character vector of the names of the instrumental variables
- data: Data to be analyzed
- controls: Character vector containing the names of the exogenous regressors
- robust: Boolean of whether to compute heteroskedasticity-robust standard errors

Value
List of beta estimates and associated standard errors for OLS and IV estimation

get_k_bounds_unrest
Given observables from the data, generates unrestricted bounds for kappa. Vectorized

Description
Given observables from the data, generates unrestricted bounds for kappa. Vectorized

Usage
get_k_bounds_unrest(obs, tilde)

Arguments
- obs: Observables generated by get_observables
- tilde: Boolean of whether or not kappa_tilde or kappa is desired

Value
List of upper bounds and lower bounds for kappa
**get_L**

*Computes L, lower bound for kappa_tilde in paper*

---

**Description**

Computes L, lower bound for kappa_tilde in paper

**Usage**

`get_L(draws)`

**Arguments**

- **draws**: data.frame of observables of simulated data

**Value**

Vector of L values

---

**get_M**

*Solves for the magnification factor*

---

**Description**

This function solves for the magnification factor given r_TstarU and kappa. It handles 3 potential cases when the magnification factor must be evaluated: 1. Across multiple simulations, but given the same r_TstarU and k 2. For multiple simulations, each with a value of r_TstarU and k 3. For one simulation across a grid of r_TstarU and k

**Usage**

`get_M(r_TstarU, k, obs)`

**Arguments**

- **r_TstarU**: Vector of r_TstarU values
- **k**: Vector of kappa values
- **obs**: Observables generated by get_observables

**Value**

Vector of magnification factors
get_new_draws  Computes beliefs that support valid instrument

Description
Computes beliefs that support valid instrument

Usage
get_new_draws(obs_draws, post_draws)

Arguments

obs_draws  data.frame of draws of reduced form parameters
post_draws  data.frame of posterior draws

Value
data.frame of new draws

get_observables  Given data and function specification, returns the relevant correlations and covariances with any exogenous controls projected out.

Description
Given data and function specification, returns the relevant correlations and covariances with any exogenous controls projected out.

Usage
get_observables(y_name, T_name, z_name, data, controls = NULL)

Arguments

y_name  Name of the dependent variable
T_name  Name(s) of the preferred regressor(s)
z_name  Name(s) of the instrumental variable(s)
data  Data to be analyzed
controls  Exogenous regressors to be included

Value
List of correlations, covariances, and $R^2$ of first and second stage regressions after projecting out any exogenous control regressors
get_psi_lower

Computes the lower bound of psi for binary data

Description
Computes the lower bound of psi for binary data

Usage
get_psi_lower(s2_T, p, kappa)

Arguments
s2_T Vector of s2_T draws from observables
p Treatment probability from binary data
kappa Vector of kappa, NOTE: kappa_tilde in the paper

Value
Vector of lower bounds for psi

get_psi_upper

Computes the upper bound of psi for binary data

Description
Computes the upper bound of psi for binary data

Usage
get_psi_upper(s2_T, p, kappa)

Arguments
s2_T Vector of s2_T draws from observables
p Treatment probability from binary data
kappa Vector of kappa, NOTE: kappa_tilde in the paper

Value
Vector of upper bounds for psi
get_p_valid

Compute the share of draws that could contain a valid instrument.

Description
Compute the share of draws that could contain a valid instrument.

Usage
get_p_valid(draws)

Arguments
draws List of simulated draws

Value
Numeric of the share of valid draws as determined by having the the restricted bounds for r_uz contain zero.

get_r_TstarU_bounds_unrest

Given observables from the data, generates the unrestricted bounds for rho_TstarU. Data does not impose any restrictions on r_TstarU Vectorized

Description
Given observables from the data, generates the unrestricted bounds for rho_TstarU. Data does not impose any restrictions on r_TstarU Vectorized

Usage
get_r_TstarU_bounds_unrest(obs)

Arguments
obs Observables generated by get_observables

Value
List of upper and lower bounds for r_TstarU
**get_r_uz**  

*Solves for r_uz given observables, r_TstarU, and kappa*

**Description**

This function solves for r_uz given r_TstarU and kappa. It handles 3 potential cases when r_uz must be evaluated: 1. Across multiple simulations, but given the same r_TstarU and kappa 2. For multiple simulations, each with a value of r_TstarU and kappa 3. For one simulation across a grid of r_TstarU and kappa.

**Usage**

```r
get_r_uz(r_TstarU, k, obs)
```

**Arguments**

- `r_TstarU`: Vector of r_TstarU values
- `k`: Vector of kappa values
- `obs`: Observables generated by get_observables

**Value**

Vector of r_uz values.

**get_r_uz_bounds**  

*Evaluates r_uz bounds given user restrictions on r_TstarU and kappa*

**Description**

This function takes observables from the data and user beliefs over the extent of measurement error (kappa) and the direction of endogeneity (r_TstarU) to generate the implied bounds on instrument validity (r_uz).

**Usage**

```r
get_r_uz_bounds(r_TstarU_lower, r_TstarU_upper, k_lower, k_upper, obs)
```

**Arguments**

- `r_TstarU_lower`: Vector of lower bounds of endogeneity
- `r_TstarU_upper`: Vector of upper bounds of endogeneity
- `k_lower`: Vector of lower bounds on measurement error
- `k_upper`: Vector of upper bounds on measurement error
- `obs`: Observables generated by get_observables
get_s_u

Solves for the variance of the error term u

Description
This function solves for the variance of u given r_TstarU and kappa. It handles 3 potential cases when the variance of u must be evaluated: 1. Across multiple simulations, but given the same r_TstarU and k 2. For multiple simulations, each with a value of r_TstarU and k 3. For one simulation across a grid of r_TstarU and k

Usage
get_s_u(r_TstarU, k, obs)

Arguments

r_TstarU  Vector of r_TstarU values
k  Vector of kappa values
obs  Observables generated by get_observables

Value
Vector of variances of u

get_r_uz_bounds_unrest

Given observables from the data, generates the unrestricted bounds for rho_uz. Vectorized

Description
Given observables from the data, generates the unrestricted bounds for rho_uz. Vectorized

Usage
get_r_uz_bounds_unrest(obs)

Arguments

obs  Observables generated by get_observables

Value
List of upper and lower bounds for rho_uz
**G function from Proposition A.2**

**Description**
G function from Proposition A.2

**Usage**
```r
g_functionA2(kappa, r_TstarU, obs_draws)
```

**Arguments**
- `kappa`: Kappa value
- `r_TstarU`: `r_TstarU` value
- `obs_draws`: a row of the data.frame of observable draws

**Value**
G value

---

**ivdoctr**

**Description**
Generates parameter estimates given user restrictions and data

**Usage**
```r
ivdoctr(
  y_name,
  T_name,
  z_name,
  data,
  example_name,
  controls = NULL,
  robust = FALSE,
  r_TstarU_restriction = c(-1, 1),
  k_restriction = c(1e-04, 1),
  n_draws = 5000,
  n_RF_draws = 1000,
  n_IS_draws = 1000,
  resample = FALSE
)
```
Arguments

- `y_name`: Character string with the column name of the dependent variable.
- `T_name`: Character string with the column name of the endogenous regressor(s).
- `z_name`: Character string with the column name of the instrument(s).
- `data`: Data frame.
- `example_name`: Character string naming estimation.
- `controls`: Vector of character strings specifying the exogenous variables.
- `robust`: Indicator for heteroskedasticity-robust standard errors.
- `r_TstarU_restriction`: 2-element vector of min and max of r_TstarU.
- `k_restriction`: 2-element vector of min and max of kappa.
- `n_draws`: Number of draws when generating frequentist-friendly draws of the covariance matrix.
- `n_RF_draws`: Number of reduced-form draws.
- `n_IS_draws`: Number of draws on the identified set.
- `resample`: Indicator of whether or not to resample using magnification factor.

Value

List with elements:

- `ols`: lm object of OLS estimation.
- `iv`: ivreg object of the IV estimation.
- `n`: Number of observations.
- `b_OLS`: OLS point estimate.
- `se_OLS`: OLS standard errors.
- `b_IV`: IV point estimate.
- `se_IV`: IV standard errors.
- `k_lower`: lower bound of kappa.
- `p_empty`: fraction of parameter draws that yield an empty identified set.
- `p_valid`: fraction of parameter draws compatible with a valid instrument.
- `r_uz_full_interval`: 90% posterior credible interval for fully identified set of rho.
- `beta_full_interval`: 90% posterior credible interval for fully identified set of beta.
- `r_uz_median`: posterior median for partially identified rho.
- `r_uz_partial_interval`: 90% posterior credible interval for partially identified set of rho under a conditionally uniform reference prior.
- `beta_median`: posterior median for partially identified beta.
- `beta_partial_interval`: 90% posterior credible interval for partially identified set of beta under a conditionally uniform reference prior.
- `a0`: If treatment is binary, mis-classification probability of no-treatment case. NULL otherwise.
• \(a_1\): If treatment is binary, mis-classification probability of treatment case. NULL otherwise
• \(\psi_{\text{lower}}\): lower bound for \(\psi\)
• \(\text{binary}\): logical indicating if treatment is binary
• \(\kappa_{\text{restriction}}\): User-specified bounds on kappa
• \(r_{T_{\text{starU}}}\) _restriction_: User-specified bounds on \(r_{T_{\text{starU}}}\)

**Examples**

```r
library(ivdoctr)
endog <- c(0, 0.9)
meas <- c(0.6, 1)

colonial_ex1 <- ivdoctr(y_name = "logpgp95", T_name = "avexpr",
z_name = "logem4", data = colonial,
controls = NULL, robust = FALSE,
r_TstarU_restriction = endog,
k_restriction = meas,
example_name = "Colonial Origins")
```

**makeTable**

Generates table of parameter estimates given user restrictions and data

**Description**

Generates table of parameter estimates given user restrictions and data

**Usage**

```r
makeTable(..., output)
```

**Arguments**

... Arguments of TeX code for individual examples to be combined into a single table

output File name to write

**Value**

LaTeX code that generates output table with regression results
Examples

```r
library(ivdoctr)
endog <- c(0, 0.9)
meas <- c(0.6, 1)

colonial_example1 <- ivdoctr(y_name = "logpgp95", T_name = "avexpr",
z_name = "logem4", data = colonial,
controls = NULL, robust = FALSE,
r_TstarU_restriction = endog,
k_restriction = meas,
example_name = "Colonial Origins")
makeTable(colonial_example1, output = file.path(tempdir(), "colonial.tex"))
```

make_full_row

*Takes the OLS and IV estimates and converts it to a row of the LaTeX table*

Description

Takes the OLS and IV estimates and converts it to a row of the LaTeX table

Usage

```r
make_full_row(stats, example_name)
```

Arguments

- `stats`: List with OLS and IV estimates and the bounds on kappa and r_uz
- `example_name`: Character string detailing the example

Value

LaTeX code passed to `makeTable()`

make_tex_row

*Makes LaTeX code to make a row of a table and shift by some amount of columns if necessary*

Description

Makes LaTeX code to make a row of a table and shift by some amount of columns if necessary

Usage

```r
make_tex_row(char_vec, shift = 0)
```
map2color

Arguments
char_vec Vector of characters to be collapsed into a LaTeX table
shift Number of columns to shift over

Value
LaTeX string of the whole row of the table

map2color Generates a custom color palette given a vector of numbers

Description
Generates a custom color palette given a vector of numbers

Usage
map2color(x, pal, limits = NULL)

Arguments
x Vector of numbers
pal Palette function generate from colorRampPalette
limits Limits on the numeric sequence

Value
Hex values for colors

myformat Rounds x to two decimal places

Description
Rounds x to two decimal places

Usage
myformat(x)

Arguments
x Number to be rounded

Value
Number rounded to 2 decimal places
plot_3d_beta  Plot ivdoctr Restrictions

Description

Plot ivdoctr Restrictions

Usage

plot_3d_beta(
  y_name,
  T_name,
  z_name,
  data,
  controls = NULL,
  r_TstarU_restriction = c(-1, 1),
  k_restriction = c(0, 1),
  n_grid = 30,
  n_colors = 500,
  fence = NULL,
  gray_k = NULL,
  gray_rTstarU = NULL,
  theta = 0,
  phi = 15
)

Arguments

y_name  Character string with the column name of the dependent variable
T_name  Character string with the column name of the endogenous regressor(s)
z_name  Character string with the column name of the instrument(s)
data  Data frame
controls  Vector of character strings specifying the exogenous variables
r_TstarU_restriction  2-element vector of bounds for r_TstarU
k_restriction  2-element vector of bounds for kappa
n_grid  Number of points to put in grid
n_colors  Number of colors to use
fence  Vector of left, bottom, right, and top corners of rectangle
gray_k  2-element vector of kappa restrictions to recolor graph as gray
gray_rTstarU  2-element vector of rTstarU restrictions to recolor graph as gray
theta  Graphing parameters for orienting plot
phi  Graphing parameters for orienting plot
**Value**

Interactive 3d plot which can be oriented and saved using `rgl.snapshot()`

**Examples**

```r
library(ivdoctr)
endog <- matrix(c(0, 0.9), nrow = 1)
meas <- matrix(c(0.6, 1), nrow = 1)

plot_3d_beta(y_name = "logpgp95", T_name = "avexpr",
             z_name = "logem4", data = colonial,
             r_TstarU_restriction = endog,
             k_restriction = meas)
```

---

**rect_points**  
*Construct vectors of points that outline a rectangle.*

**Description**

Construct vectors of points that outline a rectangle.

**Usage**

`rect_points(xleft, ybottom, xright, ytop, step_x, step_y)`

**Arguments**

- `xleft`  
  The left side of the rectangle
- `ybottom`  
  The bottom of the rectangle
- `xright`  
  The right side of the rectangle
- `ytop`  
  The top of the rectangle
- `step_x`  
  The step size of the x coordinates
- `step_y`  
  The step size of the y coordinates

**Value**

List of x-coordinates and y-coordinates tracing the points around the rectangle
**rinvwish**

*Simulate draws from the inverse Wishart distribution*

**Description**

Simulate draws from the inverse Wishart distribution

**Usage**

\[\text{rinvwish}(n, v, S)\]

**Arguments**

- \(n\) An integer, the number of draws.
- \(v\) An integer, the degrees of freedom of the distribution.
- \(S\) A numeric matrix, the scale matrix of the distribution.

**Details**

Employs the Bartlett Decomposition (Smith & Hocking 1972). Output exactly matches that of \text{rinvwish} from the MCMCpack package if the same random seed is used.

**Value**

A numeric array of matrices, each of which is one simulation draw.

---

**toList**

*Convert 3-d array to list of matrixes*

**Description**

Convert 3-d array to list of matrixes

**Usage**

\[\text{toList}(\text{myArray})\]

**Arguments**

- \(\text{myArray}\) A three-dimensional numeric array.

**Value**

A list of numeric matrices.
Becker and Woessmann (2009) Dataset

Description

Usage
weber

Format
A data frame with 452 rows and 44 variables:

- **kreiskey1871** kreiskey1871
- **county1871** County name in 1871
- **rbkey** District key
- **lat_rad** Latitude (in rad)
- **lon_rad** Longitude (in rad)
- **kmwittenberg** Distance to Wittenberg (in km)
- **zupreussen** Year in which county was annexed by Prussia
- **hhsize** Average household size
- **gpop** Population growth from 1867-1871 in percentage points
- **f_prot** Percent Protestants
- **f_jew** Percent Jews
- **f_rw** Percent literate
- **f_miss** Percent missing education information
- **f_young** Percent below the age of 10
- **f_fem** Percent female
- **f_ortsgeb** Percent born in municipality
- **f_pruss** Percent of Prussian origin
- **f_blind** Percent blind
- **f_deaf** Percent deaf-mute
- **f_dumb** Percent insane
- **f_urban** Percent of county population in urban areas
- **lnpop** Natural logarithm of total population size
- **lnkmb** Natural logarithm of distance to Berlin (km)
- **poland** Dummy variable, =1 if county is Polish-speaking
\texttt{latlon} \hspace{0.5em} \text{Latitude} \times \text{Longitude} \times 100

\texttt{f\_over3km} \hspace{0.5em} \text{Percent of pupils farther than 3km from school}

\texttt{f\_mine} \hspace{0.5em} \text{Percent of labor force employed in mining}

\texttt{inctaxpc} \hspace{0.5em} \text{Income tax revenue per capita in 1877}

\texttt{perc\_secB} \hspace{0.5em} \text{Percentage of labor force employed in manufacturing in 1882}

\texttt{perc\_secC} \hspace{0.5em} \text{Percentage of labor force employed in services in 1882}

\texttt{perc\_secBnC} \hspace{0.5em} \text{Percentage of labor force employed in manufacturing and services in 1882}

\texttt{lnyteacher} \hspace{0.5em} 100 \times \text{Natural logarithm of male elementary school teachers in 1886}

\texttt{rhs} \hspace{0.5em} \text{Dummy variable, =1 if Imperial or Hanseatic city in 1517}

\texttt{yteacher} \hspace{0.5em} \text{Income of male elementary school teachers in 1886}

\texttt{pop} \hspace{0.5em} \text{Total population size}

\texttt{kmb} \hspace{0.5em} \text{Distance to Berlin (km)}

\texttt{uni1517} \hspace{0.5em} \text{Dummy variable, =1 if University in 1517}

\texttt{reichsstadt} \hspace{0.5em} \text{Dummy variable, =1 if Imperial city in 1517}

\texttt{hansestadt} \hspace{0.5em} \text{Dummy variable, =1 if Hanseatic city in 1517}

\texttt{f\_cath} \hspace{0.5em} \text{Percentage of Catholics}

\texttt{sh\_al\_in\_tot} \hspace{0.5em} \text{Share of municipalities beginning with letter A to L}

\texttt{ncloisters1517\_pkm2} \hspace{0.5em} \text{Monasteries per square kilometer in 1517}

\texttt{school1517} \hspace{0.5em} \text{Dummy variable, =1 if school in 1517}

\texttt{dnpop1500} \hspace{0.5em} \text{City population in 1500}

\textbf{Source}

https://www.ifo.de/en/iPEHD

\textbf{References}

Index

* datasets
  - afghan, 3
  - colonial, 6
  - weber, 29

afghan, 3
b_functionA3, 4
candidate1, 4
candidate2, 5
candidate3, 5
collapse_3d_array, 6
colonial, 6
draw_bounds, 7
draw_observables, 8
draw_sigma_jeffreys, 8
format_est, 9
format_HPDI, 9
format_se, 10
g_functionA2, 21
get_alpha_bounds, 11
get_beta, 12
get_beta_bounds_binary, 12
get_beta_bounds_binary_post, 13
get_bounds_unrest, 13
get_estimates, 14
get_k_bounds_unrest, 14
get_L, 15
get_M, 15
get_new_draws, 16
get_observables, 16
get_p_valid, 18
get_psi_lower, 17
get_psi_upper, 17
get_r_TstarU_bounds_unrest, 18
get_r_uz, 19
get_r_uz_bounds, 19
get_r_uz_bounds_unrest, 20
get_s_u, 20
getCoverage, 10
getInterval, 11
ivdoctr, 21
make_full_row, 24
make_tex_row, 24
makeTable, 23
map2color, 25
myformat, 25
plot_3d_beta, 26
rect_points, 27
rinvwish, 28
toList, 28
weber, 29