

# Package: ivdoctr (via r-universe)

October 22, 2024

**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. Package based on DiTraglia and Garcia-Jimeno (2020) <[doi:10.1080/07350015.2020.1753528](https://doi.org/10.1080/07350015.2020.1753528)>.

**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](mailto:emallickhossain@gmail.com)>

**Repository** CRAN

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

## Contents

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
getCoverage . . . . .	10
getInterval . . . . .	11
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_psi_lower . . . . .	17
get_psi_upper . . . . .	17
get_p_valid . . . . .	18
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
g_functionA2 . . . . .	21
ivdoctr . . . . .	21
makeTable . . . . .	23
make_full_row . . . . .	24
make_tex_row . . . . .	24
map2color . . . . .	25
myformat . . . . .	25
plot_3d_beta . . . . .	26
rect_points . . . . .	27
rinvwish . . . . .	28
toList . . . . .	28
weber . . . . .	29

---

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

**eduhead** 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**

<https://www.jstor.org/stable/3083335>

---

b_functionA3	<i>B function from Proposition A3</i>
--------------	---------------------------------------

---

**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

---

candidate1	<i>Evaluates the corners given user bounds. Vectorized wrt multiple draws of obs.</i>
------------	---

---

**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	<i>Evaluates the edge where k is on the boundary. Vectorized wrt multiple draws of obs.</i>
------------	---

---

**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	<i>Evaluates the edge where r_TstarU is on the boundary.</i>
------------	--

---

**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	<i>Collapse 3-d array to matrix</i>
-------------------	-------------------------------------

---

**Description**

Collapse 3-d array to matrix

**Usage**

```
collapse_3d_array(myarray)
```

**Arguments**

`myarray`            A three-dimensional array.

**Value**

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

---

colonial	<i>Acemoglu, Johnson, and Robinson (2001) Dataset</i>
----------	---

---

**Description**

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

**Usage**

```
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

**loghypl** 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	<i>Computes bounds for simulated data</i>
-------------	---

---

**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**

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
r_TstarU_restriction	2 element vector of bounds on r_TstarU
k_restriction	2-element vector of bounds on kappa
n_draws	Integer number of simulations to draw

**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	<i>Simulates different data draws</i>
------------------	---------------------------------------

---

**Description**

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

**Usage**

```
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	<i>Draws covariance matrix using the Jeffrey's Prior</i>
---------------------	--

---

**Description**

Draws covariance matrix using the Jeffrey's Prior

**Usage**

```
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	<i>Creates LaTeX code for the standard error</i>
-----------	--

---

**Description**

Creates LaTeX code for the standard error

**Usage**

```
format_se(se)
```

**Arguments**

se	Standard error
----	----------------

**Value**

LaTeX string for the standard error

---

getCoverage	<i>Computes coverage of list of intervals</i>
-------------	---

---

**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	<i>Generates smallest covering interval</i>
-------------	---

---

**Description**

Generates smallest covering interval

**Usage**

```
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	<i>Computes a0 and a1 bounds</i>
------------------	----------------------------------

---

**Description**

Computes a0 and a1 bounds

**Usage**

```
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	<i>Solves for beta</i>
----------	------------------------

---

**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 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_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	<i>Returns beta bounds in binary case using grid search</i>
------------------------	---

---

**Description**

Returns beta bounds in binary case using grid search

**Usage**

```
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**

```
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**

```
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	<i>Computes OLS and IV estimates</i>
---------------	--------------------------------------

---

**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	<i>Given observables from the data, generates unrestricted bounds for kappa. Vectorized</i>
---------------------	---

---

**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	<i>Computes L, lower bound for kappa_tilde in paper</i>
-------	---

---

**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	<i>Solves for the magnification factor</i>
-------	--

---

**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	<i>Computes beliefs that support valid instrument</i>
---------------	---

---

**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	<i>Given data and function specification, returns the relevant correlations and covariances with any exogenous controls projected out.</i>
-----------------	--

---

**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	<i>Computes the lower bound of psi for binary data</i>
---------------	--

---

**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	<i>Computes the upper bound of psi for binary data</i>
---------------	--

---

**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	<i>Solves for r_uz given observables, r_TstarU, and kappa</i>
----------	---

---

**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 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_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	<i>Evaluates r_uz bounds given user restrictions on r_TstarU and kappa</i>
-----------------	--

---

**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**

```
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

**Value**

2-column data frame of lower and upper bounds of r\_uz

---

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

---

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

---

g_functionA2	<i>G function from Proposition A.2</i>
--------------	--

---

**Description**

G function from Proposition A.2

**Usage**

```
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	<i>Generates parameter estimates given user restrictions and data</i>
---------	---

---

**Description**

Generates parameter estimates given user restrictions and data

**Usage**

```
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**

<code>y_name</code>	Character string with the column name of the dependent variable
<code>T_name</code>	Character string with the column name of the endogenous regressor(s)
<code>z_name</code>	Character string with the column name of the instrument(s)
<code>data</code>	Data frame
<code>example_name</code>	Character string naming estimation
<code>controls</code>	Vector of character strings specifying the exogenous variables
<code>robust</code>	Indicator for heteroskedasticity-robust standard errors
<code>r_TstarU_restriction</code>	2-element vector of min and max of $r_{TstarU}$ .
<code>k_restriction</code>	2-element vector of min and max of $\kappa$ .
<code>n_draws</code>	Number of draws when generating frequentist-friendly draws of the covariance matrix
<code>n_RF_draws</code>	Number of reduced-form draws
<code>n_IS_draws</code>	Number of draws on the identified set
<code>resample</code>	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

- a1: If treatment is binary, mis-classification probability of treatment case. NULL otherwise
- psi\_lower: lower bound for psi
- binary: logical indicating if treatment is binary
- k\_restriction: User-specified bounds on kappa
- r\_TstarU\_restriction: User-specified bounds on r\_TstarU

## Examples

```
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	<i>Generates table of parameter estimates given user restrictions and data</i>
-----------	--

---

## Description

Generates table of parameter estimates given user restrictions and data

## Usage

```
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**

```

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	<i>Takes the OLS and IV estimates and converts it to a row of the LaTeX table</i>
---------------	---

---

**Description**

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

**Usage**

```
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	<i>Makes LaTeX code to make a row of a table and shift by some amount of columns if necessary</i>
--------------	---

---

**Description**

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

**Usage**

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



**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	<i>Generates a custom color palette given a vector of numbers</i>
-----------	---

---

**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	<i>Rounds x to two decimal places</i>
----------	---------------------------------------

---

**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**

```
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	<i>Construct vectors of points that outline a rectangle.</i>
-------------	--

---

**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

---

rinwish	<i>Simulate draws from the inverse Wishart distribution</i>
---------	---

---

**Description**

Simulate draws from the inverse Wishart distribution

**Usage**

```
rinwish(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 riwish 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	<i>Convert 3-d array to list of matrixes</i>
--------	--

---

**Description**

Convert 3-d array to list of matrixes

**Usage**

```
toList(myArray)
```

**Arguments**

myArray	A three-dimensional numeric array.
---------	------------------------------------

**Value**

A list of numeric matrices.

---

weber

*Becker and Woessmann (2009) Dataset*


---

**Description**

Data on Prussian counties in 1871 from Becker and Woessmann's (2009) paper "Was Weber Wrong? A Human Capital Theory of Protestant Economic History."

**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**hhsiz** 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

**latlon** Latitude \* Longitude \* 100  
**f\_over3km** Percent of pupils farther than 3km from school  
**f\_mine** Percent of labor force employed in mining  
**inctaxpc** Income tax revenue per capita in 1877  
**perc\_secB** Percentage of labor force employed in manufacturing in 1882  
**perc\_secC** Percentage of labor force employed in services in 1882  
**perc\_secBnC** Percentage of labor force employed in manufacturing and services in 1882  
**lnyteacher** 100 \* Natural logarithm of male elementary school teachers in 1886  
**rhs** Dummy variable, =1 if Imperial of Hanseatic city in 1517  
**yteacher** Income of male elementary school teachers in 1886  
**pop** Total population size  
**kmb** Distance to Berlin (km)  
**uni1517** Dummy variable, =1 if University in 1517  
**reichsstadt** Dummy variable, =1 if Imperial city in 1517  
**hansestadt** Dummy variable, =1 if Hanseatic city in 1517  
**f\_cath** Percentage of Catholics  
**sh\_al\_in\_tot** Share of municipalities beginning with letter A to L  
**ncloisters1517\_pkm2** Monasteries per square kilometer in 1517  
**school1517** Dummy variable, =1 if school in 1517  
**dnpop1500** City population in 1500

#### Source

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

#### References

<https://www.ifo.de/en/iPEHD> doi: 10.1162/qjec.2009.124.2.531

# 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