Package ‘desk’

September 19, 2023

Type Package

Title Didactic Econometrics Starter Kit

Version 1.1.0

Description Written to help undergraduate as well as graduate students to get started with R for basic econometrics without the need to import specific functions and datasets from many different sources. Primarily, the package is meant to accompany the German textbook Auer, L.v., Hoffmann, S., Kranz, T. (2023, ISBN: 978-3-662-68263-0) from which the exercises cover all the topics from the textbook Auer, L.v. (2023, ISBN: 978-3-658-42699-6).

URL https://github.com/OvGU-SH/desk

BugReports https://github.com/OvGU-SH/desk/issues

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports cli, rstudioapi, stats, graphics, grDevices, utils

Suggests stargazer, foreign, readxl, wooldridge, lmtest, testthat (>= 3.0.0)

Depends R (>= 3.5.0)

(Config/testthat/edition 3

NeedsCompilation no

Author Soenke Hoffmann [cre, aut],
Tobias Kranz [aut]

Maintainer Soenke Hoffmann <sohoffma@ovgu.de>

Repository CRAN

Date/Publication 2023-09-19 14:10:06 UTC
R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>acc</td>
<td>3</td>
</tr>
<tr>
<td>ar1sim</td>
<td>4</td>
</tr>
<tr>
<td>arguments</td>
<td>6</td>
</tr>
<tr>
<td>bc.model</td>
<td>7</td>
</tr>
<tr>
<td>bc.test</td>
<td>7</td>
</tr>
<tr>
<td>bp.test</td>
<td>9</td>
</tr>
<tr>
<td>cochorc</td>
<td>11</td>
</tr>
<tr>
<td>data.anscombe</td>
<td>13</td>
</tr>
<tr>
<td>data.auto</td>
<td>14</td>
</tr>
<tr>
<td>data.ballb</td>
<td>15</td>
</tr>
<tr>
<td>data.burglary</td>
<td>15</td>
</tr>
<tr>
<td>data.cars</td>
<td>16</td>
</tr>
<tr>
<td>data.cobbdoug</td>
<td>17</td>
</tr>
<tr>
<td>data.comp</td>
<td>17</td>
</tr>
<tr>
<td>data.eu</td>
<td>18</td>
</tr>
<tr>
<td>data.fertilizer</td>
<td>20</td>
</tr>
<tr>
<td>data.filter</td>
<td>21</td>
</tr>
<tr>
<td>data.govexpend</td>
<td>22</td>
</tr>
<tr>
<td>data.icecream</td>
<td>23</td>
</tr>
<tr>
<td>data.income</td>
<td>23</td>
</tr>
<tr>
<td>data.insurance</td>
<td>24</td>
</tr>
<tr>
<td>data.iv</td>
<td>25</td>
</tr>
<tr>
<td>data.lifesat</td>
<td>26</td>
</tr>
<tr>
<td>data.macro</td>
<td>26</td>
</tr>
<tr>
<td>data.milk</td>
<td>27</td>
</tr>
<tr>
<td>data.pharma</td>
<td>28</td>
</tr>
<tr>
<td>data.printer</td>
<td>29</td>
</tr>
<tr>
<td>data.regional</td>
<td>30</td>
</tr>
<tr>
<td>data.rent</td>
<td>31</td>
</tr>
<tr>
<td>data.savings</td>
<td>31</td>
</tr>
<tr>
<td>data.sick</td>
<td>32</td>
</tr>
<tr>
<td>data.software</td>
<td>33</td>
</tr>
<tr>
<td>data.spurious</td>
<td>34</td>
</tr>
<tr>
<td>data.tip</td>
<td>35</td>
</tr>
<tr>
<td>data.tip.all</td>
<td>36</td>
</tr>
<tr>
<td>data.trade</td>
<td>36</td>
</tr>
<tr>
<td>data.unempl</td>
<td>37</td>
</tr>
<tr>
<td>data.wage</td>
<td>38</td>
</tr>
<tr>
<td>data.windscreen</td>
<td>39</td>
</tr>
<tr>
<td>datasets</td>
<td>39</td>
</tr>
<tr>
<td>ddw</td>
<td>40</td>
</tr>
<tr>
<td>def.exp</td>
<td>41</td>
</tr>
<tr>
<td>def.log</td>
<td>42</td>
</tr>
<tr>
<td>dw.test</td>
<td>42</td>
</tr>
<tr>
<td>gg.test</td>
<td>44</td>
</tr>
<tr>
<td>hcc</td>
<td>46</td>
</tr>
</tbody>
</table>
**acc**

**Description**

Calculates the autocorrelation coefficient between a vector and its k-period lag. This can be used as an estimator for rho in an AR(1) process.

**Usage**

```r
acc(x, lag = 1)
```

**Arguments**

- `x`: a vector, usually residuals.
- `lag`: lag for which the autocorrelation should be calculated.
Value

Autocorrelation coefficient of lag k, numeric value.

References


See Also

lagk, acf.

Examples

```r
# Simulate AR(1) Process with 30 observations and positive autocorrelation
X <- ar1sim(n = 30, u0 = 2.0, rho = 0.7, var.e = 0.1)
acc(X$u.sim, lag = 1)

# Equivalent result using acf (stats)
acf(X$u.sim, lag.max = 1, plot = FALSE)$acf[2]
```

---

**ar1sim**

*Simulate AR(1) Process*

---

**Description**

Simulates an autoregressive process of order 1.

**Usage**

```r
ar1sim(n = 50, rho, u0 = 0, var.e = 1, details = FALSE, seed = NULL)
```

**Arguments**

- `n` number of observations to be simulated.
- `rho` true rho value of the AR(1) process to be simulated.
- `u0` start value of the process in t = 0.
- `var.e` variance of the random error. If zero, no random error is added.
- `details` logical value indicating whether details should be printed.
- `seed` optionally set a custom random seed for reproducing results.
**ar1sim**

**Value**

A list object including:

- `u.sim` vector of simulated AR(1) values.
- `n` total number of simulated AR(1) values.
- `rho` true rho value of AR(1) process.
- `e.sim` normal errors in AR(1) process.

**Note**

Objects generated by `ar1sim()` can be plotted using the regular `plot()` command. `plot.what = "time"` plots simulated AR(1) values over time. Available options are

- `...` other arguments that `plot()` understands.

`plot.what = "lag"` plots simulated AR(1) values over its lagged values. Available options are

- `true.line` logical value (default: TRUE). Should the true line be plotted?
- `acc.line` logical value (default: FALSE). Should the autocorrelation coefficient line be plotted?
- `ols.line` logical value (default: FALSE). Should the ols regression line be plotted?
- `...` other arguments that `plot()` understands.

**Examples**

```r
## Generate 30 positively autocorrelated errors
my.ar1 <- ar1sim(n = 30, rho = 0.9, var.e = 0.1, seed = 511)
my.ar1
plot(my.ar1$u.sim, type = 'l')
## Illustrate the effect of Rho on the AR(1)
set.seed(12)
parOrg = par(c("mfrow", "mar"))
par(mfrow = c(2,4), mar = c(1,1,1,1))
rhovalues <- c(0.1, 0.5, 0.8, 0.99)
for (i in c(0, 0.3)){
  for (rho in rhovalues){
    u.data <- ar1sim(n = 20, u0 = 2, rho = rho, var.e = i)
    plot(u.data$u.sim, plot.what = "lag", cex.legend = 0.7, xlim = c(-2.5,2.5), ylim = c(-2.5,2.5),
         acc.line = TRUE, ols.line = TRUE)
  }
}
par(mfrow = parOrg$"mfrow", mar = parOrg$"mar")
## Illustrate the effect of Rho on the (non-)stationarity of the AR(1)
set.seed(1324)
parOrg = par(c("mfrow", "mar"))
par(mfrow = c(2, 4), mar = c(1,1,1,1))
```

for (rho in c(0.1, 0.9, 1, 1.04, -0.1, -0.9, -1, -1.04)) {
  u.data <- ar1sim(n = 25, u0 = 5, rho = rho, var.e = 0)
  plot(u.data$u.sim, plot.what = "time", ylim = c(-8,8))
}
par(mfrow = parOrg$"mfrow", mar = parOrg$"mar")

---

**Arguments**

**Arguments of a Function**

**Description**

Shows the arguments and their default values of a function.

**Usage**

```r
arguments(fun, width = options("width")$width)
```

**Arguments**

- `fun` name of the function.
- `width` optional width for line breaking.

**Value**

None.

**See Also**

`args`.

**Examples**

```r
arguments(repeat.sample)
```
bc.model  

One Dimensional Box-Cox Model

Description

Finds lambda-values for which the one dimensional Box-Cox model has lowest SSR.

Usage

bc.model(mod, data = list(), range = seq(-2, 2, 0.1), details = FALSE)

Arguments

mod  
estimated linear model object or formula.

data  
if mod is a formula then the corresponding data frame has to be specified.

range  
range and step size of lambda values. Default is a range from -2 to 2 at a step size of 0.1.

details  
logical value indicating whether specific details about the test should be returned.

Value

A list object including:

results  
regression results with minimal SSR.

lambda  
optimal lambda-values.

nregs  
o. of regressions performed.

idx.opt  
index of optimal regression.

val.opt  
minimal SSR value.

Examples

y <- c(4,1,3)
x <- c(1,2,4)
my.mod <- ols(y ~ x)
bc.model(my.mod)

bc.test  
Box-Cox Test

Description

Box-Cox test for functional form. Compares a base model with non transformed endogenous variable to a model with logarithmic endogenous variable. Exogenous variables can be transformed
or non-transformed. The object of test results returned by this command can be plotted using the plot() function.

Usage

```r
bc.test(
  basemod,
  data = list(),
  exo = "same",
  sig.level = 0.05,
  details = TRUE,
  hyp = TRUE
)
```

Arguments

- **basemod**: estimated linear model object or formula taken as the base model for comparison. Has to have a non-transformed endogenous variable.
- **data**: if `mod` is a formula then the corresponding data frame has to be specified.
- **exo**: vector or matrix of transformed exogenous variables to be used in the comparison model. If not specified the same variables from the base model are used ("same").
- **sig.level**: significance level. Default value: `sig.level = 0.05`.
- **details**: logical value indicating whether specific details about the test should be returned.
- **hyp**: logical value indicating whether the Hypotheses should be returned.

Value

A list object including:

- **hyp**: character matrix of hypotheses (if `hyp = TRUE`).
- **results**: a data frame of basic test results.
- **stats**: additional statistic of aux. regression.
- **nulldist**: type of the Null distribution with its parameters.

References


See Also

`boxcox`.

Examples

```r
## Box-Cox test between a semi-logarithmic model and a logarithmic model
```
semilogmilk.est <- ols(milk ~ log(feed), data = data.milk)
results <- bc.test(semilogmilk.est, details = TRUE)

## Plot the test results
plot(results)

## Example with transformed exogenous variables
lin.est <- ols(rent ~ mult + mem + access, data = data.comp)
A <- lin.est$data
bc.test(lin.est, exo = log(cbind(A$mult, A$mem, A$access)))

---

**bp.test**

**Breusch-Pagan Test**

**Description**

Breusch-Pagan test for heteroskedastic errors. The object of test results returned by this command can be plotted using the `plot()` function.

**Usage**

```r
bp.test(
  mod,
  data = list(),
  varmod = NULL,
  koenker = TRUE,
  sig.level = 0.05,
  details = FALSE,
  hyp = TRUE
)
```

**Arguments**

- `mod`: estimated linear model object or formula.
- `data`: if `mod` is a formula then the corresponding data frame has to be specified.
- `varmod`: formula object (starting with tilde ~) specifying the terms of regressors that explain sigma squared for each observation. If not specified the regular model `mod` is used.
- `koenker`: logical value specifying whether Koenker’s studentized version or the original Breusch-Pagan test should be performed.
- `sig.level`: significance level. Default value: `sig.level = 0.05`.
- `details`: logical value indicating whether specific details about the test should be returned.
- `hyp`: logical value indicating whether the Hypotheses should be returned.
Value

List object including:
## BP test with Koenker's studentized residuals

```r
X <- bp.test(wage ~ educ + age, data = data.wage, koenker = FALSE)
X
```

## A white test for the same model (auxiliary regression specified by \code{varmod})

```r
bp.test(wage ~ educ + age, varmod = (educ + age)^2 + I(educ^2) + I(age^2), data = data.wage)
```

## Similar test

```r
wh.test(wage ~ educ + age, data = data.wage)
```

## Plot the test result

```r
plot(X)
```

---

### cochorc

**Estimating Linear Models under AR(1) with Cochrane-Orcutt Iteration**

**Description**

If autocorrelated errors can be modeled by an AR(1) process (rho as parameter) then this function performs a Cochrane-Orcutt iteration. If model coefficients and the estimated rho value converge with the number of iterations, this procedure provides valid solutions. The object returned by this command can be plotted using the \texttt{plot()} function.

### References


### See Also

\texttt{wh.test}, \texttt{bptest}.
cochorc

Usage

cochorc(
  mod,
  data = list(),
  iter = 10,
  tol = 0.001,
  pwt = TRUE,
  details = FALSE
)

Arguments

  mod          estimated linear model object or formula.
  data         data frame to be specified if mod is a formula.
  iter         maximum number of iterations to be performed.
  tol          iterations are carried out until difference in rho values is not larger than tol.
  pwt          build first observation using Prais-Whinston transformation. If pwt = FALSE then the first observation is dropped, Default value: pwt = TRUE.
  details      logical value, indicating whether details should be printed.

Value

A list object including:

  results      data frame of iterated regression results.
  niter        number of iterated regressions performed.
  rho.opt      rho-value at last iteration performed.
  y.trans      transformed y-values at last iteration performed.
  X.trans      transformed x-values (incl. z) at last iteration performed.
  resid        residuals of transformed model estimation.
  all.regs     data frame of regression results for all considered rho-values.

References


Examples

## In this example only 2 iterations are needed to achieve (convergence of rho at the 5th digit)
sales.est <- ols(sales ~ price, data = data.filter)
cochorc(sales.est)

## For a higher precision we need 6 iterations
cochorc(sales.est, tol = 0.0000000000001)

## Direct usage of a model formula
data.anscombe

X <- cochorc(sick ~ jobless, data = data.sick[1:14,], details = TRUE)

## See iterated regression results
X$all.regs

## Print full details
X

## Suppress details
print(X, details = FALSE)

## Plot rho over iterations to see convergence
plot(X)

## Example with interaction
dummy <- as.numeric(data.sick$year >= 2005)
kstand.str.est <- ols(sick ~ dummy + jobless + dummy*jobless, data = data.sick)
cochorc(kstand.str.est)

---

data.anscombe  
*Anscombe's Quartet*

**Description**

This data set comprises four individual x-y-data sets which have the same statistical properties (mean, variance, correlation, regression line, etc.), yet are quite different.

**Usage**

data.anscombe

**Format**

A data frame of 4 data sets, each with 11 observations of the two variables x and y.

- x1 to x4  x-variables of the four data sets.
- y1 to y4  y-variables of the four data sets.

**Details**

In Auer et al. (2023, Chap. 3) these data are used to illustrate the simple regression model and the importance to visually evaluate datasets before a numerical analysis is performed.

**Source**

This dataset was manually generated from: Anscombe, F.J. (1973): Graphs in Statistical Analysis. American Statistician, 27(1), 17-21. Also available in the R package **datasets**.
References

data.auto

Description
This is a data set on the prices and qualitative characteristics of US-cars sold in 1979.

Usage
data.auto

Format
A data frame with 52 observations on the following nine variables:

- make
- price
- mpgall
- headroom
- trunk
- weight
- length
- turn
- displacement

make make and model.
price price (in dollar).
mpgall mileage (miles per gallon).
headroom headroom (in inch)./trunk trunk Space (in cubic foot).
weight weight (in pound).
length length (in inch).
turn turn circle (in foot).
displacement displacement (in cubic inch).

Details
In Auer et al. (2023, Chap. 13) these data are used to illustrate the selection process of exogenous variables.

Source
This data frame was imported from an SAS dataset provided by York University, CA

References
**data.ballb**  
*Defective Ball Bearings*

**Description**  
This is a data set on the percentage of defective units in the production of ball bearings.

**Usage**  
data.ballb

**Format**  
A data frame with seven observations on the following two variables:

- `defbb` share of defective ball bearings (per thousand).
- `nshifts` number of shifts between two maintenances.

**Details**  
In Auer (2023, Chap. 16) and Auer et al. (2023, Chap. 16) these hypothetical data are used to illustrate the consequences of error terms with an expected value deviating from zero.

**Source**  

**References**  

---

**data.burglary**  
*Burglaries and Power Blackouts*

**Description**  
This is a data set on the monthly number of burglaries and the number of power blackouts in a small town.

**Usage**  
data.burglary

---
Format

A data frame with 12 observations on the following three variables:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>month.</td>
</tr>
<tr>
<td>burglary</td>
<td>number of burglaries.</td>
</tr>
<tr>
<td>blackout</td>
<td>number of power blackouts.</td>
</tr>
</tbody>
</table>

Details

In Auer et al. (2023, Chap. 15) these hypothetical data are used to illustrate the consequences of a structural break.

Source


---

data.cars  
*Speed and Stopping Distances of Cars*

Description

The data give the speed of cars and the distances taken to stop. The data were recorded in the 1920s.

Usage
data.cars

Format

A data frame of 50 observations with the following two variables:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>speed</td>
<td>speed (in miles per hour).</td>
</tr>
<tr>
<td>dist</td>
<td>stopping distance (in foot).</td>
</tr>
</tbody>
</table>

Details

In Auer et al. (2023, Chaps. 5, 6, 7 & 16) the data are used to illustrate the simple regression model and the consequences of truncated data.

Source

data.comp

References

data.cobbdoug

Cobb-Douglas Production Function

Description
This data set can be used to model a Cobb-Douglas production process.

Usage
data.cobbdoug

Format
A data frame with 100 observations on the following three variables:

- output: production output.
- labor: input of labor.
- capital: input of capital.

Details
In Auer et al. (2023, Chap. 14) these hypothetical data are used to illustrate the functional specification of a non-linear regression model.

Source

References

data.comp

Monthly Rentals and Qualitative Characteristics of Computers

Description
This is a data set on the monthly rentals of computers of different quality during the 1960s.
Usage
data.comp

Format
A data frame with 34 observations on the following four variables:

- **rent**: monthly rental (in dollar).
- **mem**: memory capacity computed from three different computer characteristics.
- **access**: average time required to access information from memory.
- **mult**: average time required to obtain and complete multiplication instruction.

Details
In Auer et al. (2023, Chaps. 13 & 14) these data are used to illustrate the specification of a multivariate regression model.

Source
The dataset was originally published by Chow (1967). For the purpose of desk it was imported from 3.5 inch floppy disk in ASCII format included in Berndt (1990). The dataset also available in the original format on Github.

References

---

**data.eu**

*Expenditures of the EU-25*

Description
This is a data set on the shares of total EU-expenditures received by the individual member states of the EU-25 in 2005. Furthermore, the data describe some relevant characteristics (population share, gross domestic product, etc.) of these member states.

Usage
data.eu
Format

A data frame with 25 observations on the following seven variables:
data.fertilizer

Description

This is a data set on the use of fertilizers (phosphate and nitrogen) in the cultivation of barley.

Usage

data.fertilizer

Format

A data frame with 30 observations on the following three variables:

- phos amount of phosphate (in kg per hectare).
- nit amount of nitrogen (in kg per hectare).
- barley barley crop yield (in units of 100 kg per hectare).

Details

In Auer (2023, Chap. 9) and Auer et al. (2023, Chap. 9). These hypothetical data are used to illustrate the estimation of a multivariate linear regression model.
data.filter

Source

References

---

Water Filter Sales

Description
This is a data set on the prices and sales figures of water filters (in 1000 pcs.).

Usage
data.filter

Format
A data frame with 24 observations on the following two variables:

<table>
<thead>
<tr>
<th>variable</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sales</td>
<td>monthly water filter sales (in 1000 pcs.)</td>
</tr>
<tr>
<td>price</td>
<td>price (in Euro)</td>
</tr>
</tbody>
</table>

Details
In Auer (2023, Chap. 18) and Auer et al. (2023, Chap. 18) these hypothetical data are used to illustrate the consequences of autocorrelated error terms.

Source

References

This is a data set on the yearly expenditures of the US-States in 2013. Furthermore, the data describe some relevant characteristics of these states.

A data frame with 50 observations on the following 5 variables:

- **state**: name of the state.
- **exp**: total state expenditures per capita (in dollar).
- **aid**: federal aid received by this state (in million dollar).
- **gdp**: gross domestic product (in million dollar).
- **pop**: population (in million).

In Auer et al. (2023, Chap. 17) these data are used to illustrate the consequences of heteroscedastic error terms.

Different datasets based on National Association of State imported in 2015:

- State Expenditure Report, Table 1: Total State Expenditures - Capital Inclusive from (Budget Officers).
- Annual Surveys of State and Local Government Finances, Table 1: State and Local Government Finances by Level of Government and by State 2012-13 from U.S. Census.
- Real GDP by State, 2011-2014, Table 1 from U.S. Bureau of Economic Analysis.
- Annual Estimates of the Resident Population for the United States, Regions, States, and Puerto Rico, Table 1 from U.S. Census.

data.icecream  

_Sales of Ice Cream_

**Description**

This hypothetical data set is on the daily revenues from selling ice cream and the daily average temperature in some town on a sample of 35 working days.

**Usage**

data.icecream

**Format**

A data frame with 35 observations on the following two variables:

- `revenue` revenues (in Euro).
- `temp` temperature (in degree Celsius).

**Details**

In Auer et al. (2023, Chap. 7) these hypothetical data are used to illustrate the estimation of the simple linear regression model.

**Source**


---

data.income  

_Income Per Capita_

**Description**

This data set describes major macroeconomic variables determining the differences in per capita income of 75 countries in 1985.

**Usage**

data.income

**Format**

A data frame with 75 observations on the following three variables:
Details

In Auer (2023, Chap. 19) and Auer et al. (2023, Chap. 19) these data are used to illustrate the detection and consequences of error terms that are not normally distributed.

Source


References


<table>
<thead>
<tr>
<th>data.insurance</th>
<th>Sales of Insurance Contracts</th>
</tr>
</thead>
</table>

Description

This is a data set on the ability and success of salespersons in selling insurance contracts.

Usage

data.insurance

Format

A data frame with 30 observations on the following four variables:

- `contr` number of insurance contracts currently sold by the salesperson.
- `score` score of salesperson in assessment center.
- `contrprev` number of insurance contracts sold period by the salesperson in the previous.
- `ability` salesperson’s true ability to sell insurance contracts.
Details

In Auer (2023, Chap. 20) and Auer et al. (2023, Chap. 20) these hypothetical data illustrate the use of two stage least squares estimation with an instrumental variable.

Source


References


<table>
<thead>
<tr>
<th>data.iv</th>
<th>Instrumental Variables</th>
</tr>
</thead>
</table>

Description

This data set is on the use of instrumental variables.

Usage

data.iv

Format

A data frame with 8 observations on the following five variables:

- y  endogenous variable.
- x1 first exogenous variable.
- x2 second exogenous variable.
- z1 first instrumental variable.
- z2 second instrumental variable.

Details

In Auer et al. (2023, Chap. 20) these hypothetical data are used to illustrate the use of two stage least squares estimation with instrumental variables.

Source

data.lifesat  

*Life Satisfaction*

**Description**

A data set describing the life satisfaction and per capita income in 40 countries in 2010.

**Usage**

data.lifesat

**Format**

A data frame of 40 observations with the following three variables:

- country: country name.
- income: country’s per capita income (in dollar).
- lsat: index of country’s average life satisfaction.

**Details**

In Auer et al. (2023, Chap. 3) these hypothetical data are used to illustrate the use of the simple linear regression model.

**Source**

Imported from *World Value Survey, Inglehart et al. (2014)*.

**References**


---

data.macro  

*Macroeconomic Data from Germany*

**Description**

This is a (time series) data set on macroeconomic data from Germany covering 129 consecutive quarters.
**data.milk**

**Usage**

data.macro

**Format**

A data frame with 129 observations on the following seven variables:

- `quarter`: identifies the time period in combination with year.
- `year`: identifies the time period in combination with quarter.
- `consump`: private consumption in the observed quarter.
- `invest`: gross investment in the observed quarter.
- `gov`: government expenditure in the observed quarter.
- `netex`: net exports (exports - imports) in the observed quarter.
- `gdp`: gross domestic product in the observed quarter.

**Details**

These National Accounts data are measured in real quantities (billions of chained 2015 Euros) and are calendar and seasonally-adjusted (method: X13 JDemetra+). Theoretically, private consumption, gross investment, government expenditure, and net exports should exactly sum up to the gross domestic product. However, in practice, there are often some minor discrepancies in the data. As a result, for didactical purposes, we calculated gross investment as residuals rather than using the actual data.

**Source**


**References**


---

**data.milk**

*Milk Production*

**Description**

This is a hypothetical data set on the use of concentrated feed for cows and their milk output.

**Usage**

data.milk

**Format**

A data frame with 12 observations on the following two variables:
Details

In Auer (2023, Chap. 14) and Auer et al. (2023, Chap. 14) these hypothetical data are used to illustrate transformations in non-linear relationships.

Source


References


---

**data.pharma**  
*Pharmaceutical Advertisements*

**Description**

This is a data set on quarterly commercials data of a pharmaceutical company.

**Usage**

data.pharma

**Format**

A data frame with 24 quarterly observations on the following four variables:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sales</td>
<td>sales of pharmaceutical product (in units of 100g).</td>
</tr>
<tr>
<td>ads</td>
<td>number of advertisements (in double pages).</td>
</tr>
<tr>
<td>price</td>
<td>price of pharmaceutical product (in euro per 100g).</td>
</tr>
<tr>
<td>adsprice</td>
<td>price of advertisements (in units of 1000 euro per double page).</td>
</tr>
</tbody>
</table>

**Details**

In Auer (2023, Chap. 23) and Auer et al. (2023, Chap. 23) these hypothetical data are used to illustrate the estimation of simultaneous equation econometric models.
**data.printer**

**Sources**


**References**


---

**Description**

This is a data set on the prices and qualitative characteristics of laser printers from 1992 to 2001.

**Usage**

data.printer

**Format**

A data frame with 44 observations on the following five variables:

- **price**: price of the printer (in euro).
- **speed**: printer’s speed (in pages per minute).
- **size**: printer’s size (in cubic decimeter).
- **mcost**: maintenance costs of printer (in cent per page).
- **tdiff**: time difference between the printer’s observation and the data set’s first observed laser printer (in month).

**Details**

In Auer (2023, Chap. 21) and Auer et al. (2023, Chap. 21) these hypothetical data are used to illustrate the consequences of multicollinear exogenous variables.

**Source**

Data from computer magazin c’t (February 1992 to August 2001).


**References**

Description

This is a data set on regional wages and the regional levels of the cost of living. The data set covers the 401 counties and cities of Germany.

Usage

data.regional

Format

A data frame with 401 observations on the following seven variables:

- id: identifies the region.
- region: the German name of the region.
- area: the region’s area (in square kilometers).
- pop: the region’s population in 2019.
- coli: the region’s index number of the cost of living in May 2019 (German average = 100).
- wage: the region’s median wage in December 2016 (in euro).
- unempl: the region’s unemployment rate in December 2016 (in percent).

Details

In Auer et al. (2023, Chap. 22) these data are used to illustrate the estimation of simultaneous equations models.

Source

The wage data are taken from Fuchs (2018) while the cost of living data are taken from Auer and Weinand (2022). The unemployment data can be found in the report "Arbeitsmarkt in Zahlen" provided by the Bundesagentur für Arbeit. For each German State and each month, one report is published. Each report is available as Excel-sheet.

References


---

**data.rent**

*Average Basic Rent in City Districts*

**Description**

This is a hypothetical data set on twelve districts of a city. The data describe the district’s distance to the city center and the average basic rent (it excludes additional costs).

**Usage**

data.rent

**Format**

A data frame with 12 observations on the following four variables:

- **rent**: district’s basic rent (in euro per square meter).
- **dist**: distance between district and city center (in km).
- **share**: share of rental properties considered for random selection.
- **area**: usable area (in square meter).

**Details**

In Auer (2023, Chap. 17) and Auer et al. (2023, Chap. 17) these hypothetical data are used to illustrate the consequences of heteroskedastic error terms.

**Source**


**References**


---

**data.savings**

*International Life-Cycle Savings and Disposable Income*
Description

This data set describes the savings behavior of 50 countries in 1960-1970. The data set includes demographical variables as well as variables on disposable income.

Usage

data.savings

Format

A data frame with 50 observations on the following five variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sr</td>
<td>ratio of the country’s private savings to its disposable income.</td>
</tr>
<tr>
<td>pop15</td>
<td>share of the country’s population under 15.</td>
</tr>
<tr>
<td>pop75</td>
<td>share of the country’s population over 75.</td>
</tr>
<tr>
<td>dpi</td>
<td>country’s real per capita disposable income (in dollar).</td>
</tr>
<tr>
<td>ddpi</td>
<td>growth rate of the country’s disposable income per capita (in percent).</td>
</tr>
</tbody>
</table>

Details

Under the life-cycle savings hypothesis as developed by Franco Modigliani, the savings ratio (aggregate personal saving divided by disposable income) is explained by per-capita disposable income, the percentage rate of change in per-capita disposable income, and two demographic variables: the percentage of population less than 15 years old and the percentage of the population over 75 years old. The data are averaged over the decade 1960-1970 to remove the business cycle or other short-term fluctuations.

In Auer et al. (2023, Chaps. 9, 10 & 12) the data set is used to illustrate the econometric analysis of a multivariate linear regression model.

Source

R package datasets (object LifeCycleSavings).

References


---

data.sick

Sick Leave and Unemployment

Description

This is a data set on the unemployment rates and the sick leave in Germany in the years 1992 to 2014.
Usage
data.sick

Format
A data frame with 23 observations on the following three variables:

<table>
<thead>
<tr>
<th>year</th>
<th>year.</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobless</td>
<td>average unemployment rate during that year (in percent).</td>
</tr>
<tr>
<td>sick</td>
<td>average of employees’ sick leave during that year (in percent).</td>
</tr>
</tbody>
</table>

Details
In Auer et al. (2023, Chap. 18) these data are used to illustrate the consequences of autocorrelated error terms.

Source
Imported from Federal Statistical Office of Germany.

References

Description
This is a hypothetical (time series) data set on business data of a software company covering 36 consecutive months.

Usage
data.software

Format
A data frame with 36 observations on the following three variables:

<table>
<thead>
<tr>
<th>period</th>
<th>identifies the time period.</th>
</tr>
</thead>
<tbody>
<tr>
<td>empl</td>
<td>number of employees in the observed month.</td>
</tr>
<tr>
<td>orders</td>
<td>number of new orders during the observed month.</td>
</tr>
</tbody>
</table>
**Details**

In Auer (2023, Chap. 22) and Auer et al. (2023, Chap. 22) these hypothetical data are used to illustrate the estimation of dynamic regression models.

**Source**


**References**


---

**data.spurious**  
*Non-Stationary Time Series Data*

**Description**

The variables in this data set are non-stationary and help to understand spurious regression in the context of time series analysis.

**Usage**

data.spurious

**Format**

A data frame with yearly observations from 1880 to 2022 on the following five variables:

- **year**: year of the observation.
- **temp**: deviation of the pre-industrial average global temperature.
- **elements**: number of discovered elements in chemistry (periodic table).
- **gold**: price for 1 ounce of fine gold in US-Dollar (not inflation-adjusted) starting in 1968.
- **cpi**: consumer price index: total all items for the United States (index 2015 = 100) starting in 1968.

**Source**

LBMA (retrieved from Deutsche Bundesbank Zeitreihen-Datenbanken, BBEX3.A.XAU.USD.EA.AC.C08).
OECD (retrieved from FRED, https://fred.stlouisfed.org/series/CPALTT01USA661S).

References


data.tip

Tip Data in a Restaurant

Description

This is a data set on the bills and the corresponding tips given in a restaurant of only 3 guests. Is can be used as minimal example to illustrate simple linear regression. The larger version of this dataset (20 guests) is available as data.tip.all.

Usage

data.tip

Format

A data frame with three observations on the following two variables:

- x the guest’s bill (in euro).
- y the tip given to the waiter/waitress (in euro).

Details

In Auer (2023, Chap. 17) and Auer et al. (2023, Chap. 17) these hypothetical data provide a minimal data set for estimating a simple linear regression model.

Source


References

**data.tip.all**

*Tip Data in a Restaurant with all 20 observations. Only used in textbook.*

**Description**

This is a hypothetical data set on the bills and the corresponding tips given in a restaurant. A reduced version of this dataset (only 3 observations) is also available as *data.tip*.

**Usage**

`data.tip`

**Format**

A data frame with 20 observations on the following two variables:

- `x` the guest’s bill (in euro).
- `y` the tip given to the waiter/waitress (in euro).

**Details**

In Auer (2023, Chap. 17) and Auer et al. (2023, Chap. 17) these hypothetical data provide a data set for estimating a simple linear regression model.

**Source**


**References**


---

**data.trade**

*Gravity Model Applied to Germany*

**Description**

This is a data set on German trade with its 27 EU-partners in 2014.

**Usage**

`data.trade`
**Format**

A data frame with 27 observations on the following five variables:

- **country**: name of member state.
- **imports**: German imports from member state (in million euro).
- **exports**: German exports to member state (in million euro).
- **gdp**: gross domestic product of member state (in million euro).
- **dist**: distance between member state and Germany (in km).

**Details**

In Auer et al. (2023, Chaps. 9 & 14) these data are used to illustrate the estimation and functional specification of a multivariate linear regression model.

**Source**

Imported from Eurostat Eurostat. Distances computed with FreeMapTools.

**References**


---

**data.unempl**

*German Economic Growth and Unemployment Rates*

**Description**

This is a data set on German economic growth and unemployment rates from 1992 to 2021.

**Usage**

data.unempl

**Format**

A data frame with 30 observations on the following three variables:

- **year**: year.
- **unempl**: change in German unemployment rate (in percentage points).
- **gdp**: change in German gross domestic product (in percentage).
Details

In Auer (2023, Chap. 15) and Auer et al. (2023, Chap. 15) these yearly data are used to illustrate the estimation of regression models that exhibit a structural break.

Source

Imported from Genesis, Federal Statistical Office of Germany.


References


data.wage

Wage Data in a Company

Description

This is a data set on the wage structure in a company.

Usage

data.wage

Format

A data frame with 20 observations on the following six variables:

<table>
<thead>
<tr>
<th>variable</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wage</td>
<td>employee’s monthly wage (in euro).</td>
</tr>
<tr>
<td>educ</td>
<td>employee’s extra education beyond the basic schooling degree (in years).</td>
</tr>
<tr>
<td>age</td>
<td>employee’s age (in years).</td>
</tr>
<tr>
<td>emp1</td>
<td>employee’s time of employment in the company (in years).</td>
</tr>
<tr>
<td>score</td>
<td>employee’s IQ test score.</td>
</tr>
<tr>
<td>sex</td>
<td>employee’s sex (0 = male).</td>
</tr>
</tbody>
</table>

Details

In Auer (2023, Chap. 13) and Auer et al. (2023, Chap. 13) these hypothetical data are used to illustrate the selection of the relevant exogenous variables.

Source

**References**


### data.windscreen

**Efficiency of a Car Glass Service Company**

**Description**

This is a data set on the business statistics of 248 branches of a car glass service company in 2015.

**Usage**

data.windscreen

**Format**

A data frame with 248 observations on the following eight variables:

- `screen`: number of windscreen replacements in the branch.
- `foreman`: foremen employed in the branch.
- `assist`: assistants employed in the branch.
- `f_wage`: foremen’s average wage in the branch.
- `a_wage`: assistants’ average wage in the branch.
- `f_age`: foremen’s average age in the branch.
- `a_age`: assistants’ average age in the branch.
- `capital`: total value of machines used for windscreen replacement in the branch (in euro).

**Details**

In Auer et al. (2023, Chap. 20) these hypothetical data illustrate the use of two stage least squares estimation with instrumental variables.

**Source**


### datasets

**Datasets in DESK**

**Description**

Generates a table of data set names and descriptions available in package desk.
Usage

datasets()

Value

An object of class table.

Examples

datasets()

\begin{itemize}
\item \textit{ddw} \textit{Distribution}
\end{itemize}

Description


Usage

ddw(x, mod, data = list())

Arguments

\begin{itemize}
\item \texttt{x} \quad quantile value(s) at which the density should be determined.
\item \texttt{mod} \quad estimated linear model object, formula (with argument \texttt{data} specified), or model matrix.
\item \texttt{data} \quad if \texttt{mod} is a formula then the name of the corresponding dataframe has to be specified here.
\end{itemize}

Details

The Durbin Watson Null-Distribution depends on values of the exogenous variables. That is why it must be calculated from each specific data set, respectively.

Value

Numerical density value(s).

References


def.exp

See Also
dw.test.pdw.

Examples
filter.est <- ols(sales ~ price, data = data.filter)
ddw(x = c(0.9, 1.7, 2.15), filter.est)

---

**def.exp**  
*Lambda Deformed Exponential*

Description
Calculates the lambda deformed exponential.

Usage
def.exp(x, lambda = 0, normalize = FALSE)

Arguments
- **x**: a numeric value.
- **lambda**: deformation parameter. Default value: lambda = 0 (regular exponential).
- **normalize**: logical value to indicate normalization.

Value
The function value of the lambda deformed exponential at x.

See Also
def.log.

Examples
def.exp(3)  # Natural exponential of 3
def.exp(3,2) # Deformed by lambda = 2
def.log  

*Lambda Deformed Logarithm*

**Description**

Calculates the lambda deformed logarithm.

**Usage**

```r
def.log(x, lambda = 0, normalize = FALSE)
```

**Arguments**

- `x`: a numeric value.
- `lambda`: deformation parameter. Default value: `lambda = 0` (natural log).
- `normalize`: normalization (internal purpose).

**Value**

The function value of the lambda deformed logarithm at `x`.

**See Also**

- `def.exp`.

**Examples**

```r
def.log(3)  # Natural log of 3
def.log(3, 2)  # Deformed by lambda = 2
```

---

dw.test  

*Durbin-Watson Test on AR(1) Autocorrelation*

**Description**

Durbin-Watson Test on AR(1) autocorrelation of errors in a linear model. The object of test results returned by this command can be plotted using the `plot()` function.
dw.test

Usage

\texttt{dw.test(}
  \texttt{mod,}
  \texttt{data = list(),}
  \texttt{dir = c("left", "right", "both"),}
  \texttt{method = c("pan1", "pan2", "pao1", "spa"),}
  \texttt{crit.val = TRUE,}
  \texttt{sig.level = 0.05,}
  \texttt{details = FALSE,}
  \texttt{hyp = TRUE}
\texttt{)}

Arguments

\begin{itemize}
  \item \texttt{mod} estimated linear model object or formula describing the model.
  \item \texttt{data} if \texttt{mod} is a formula then the corresponding data frame has to be specified.
  \item \texttt{dir} direction of the alternative hypothesis: "right" for $\rho > 0$, "left" for $\rho < 0$ and "both" for $\rho \neq 0$.
  \item \texttt{method} algorithm used to calculate the p-value. "pan1" and "pan2" are two implementations of Imhof's (1961) algorithm. If they provide a p-values, it is the exact one. "pao1" is Paolella's (2007) re-implementation of Imhof's theory, "spa" is a saddle point approximation, also implemented by Paolella (2007).
  \item \texttt{crit.val} logical value indicating whether the critical value should be calculated.
  \item \texttt{sig.level} significance level. Default value: \texttt{sig.level = 0.05}.
  \item \texttt{details} logical value indicating whether specific details about the test should be returned.
  \item \texttt{hyp} logical value indicating whether the Hypotheses should be returned.
\end{itemize}

Value

A list object including:

\begin{itemize}
  \item \texttt{hyp} character matrix of hypotheses (if \texttt{hyp = TRUE}).
  \item \texttt{results} a data frame of basic test results, including critical- and p-value.
  \item \texttt{nulldist} type of the null distribution (for internal use).
\end{itemize}

References


See Also

\texttt{ddw, pdw}.
Examples

```
## Estimate a simple model
filter.est <- ols(sales ~ price, data = data.filter)

## Perform Durbin Watson test for positive autocorrelation rho > 0 (i.e. d < 2)
test.results <- dw.test(filter.est)

## Print the test results
test.results

## Calculate DW null-distribution and plot the test results
plot(test.results)
```

---

gq.test  

*Goldfeld-Quandt Test*

Description

Goldfeld-Quandt test for heteroskedastic errors. The object of test results returned by this command can be plotted using the `plot()` function.

Usage

```
gq.test(
  mod,  
data = list(),
split = 0.5,  
omit.obs = 0,  
ah = c("increasing", "unequal", "decreasing"),
order.by = NULL,
sig.level = 0.05,
details = FALSE,
hyp = TRUE
)
```

Arguments

- `mod`  
  estimated linear model object or formula. If only a model formula is passed then the data argument must be specified.

- `data`  
  if mod is a formula then the corresponding data frame has to be specified.

- `split`  
  partitions the data set into two groups. If <= 1 then split is a percentage value such that $T \times \text{split}$ observations are in the first partition. If split >= 1 it is interpreted as the index of the partitioning observation, i.e. the number of observations in the first group.
omit.obs

the number of central observations to be omitted. Might increase the power of
the test. If <= 1 then split is the percentage value of all observations, otherwise
it is interpreted as absolute number.

ah

character string specifying the type of the alternative hypothesis: "increasing"
(variance increases from group 1 to group 2), "decreasing" (variance decreases
from group 1 to group 2), "unequal" (variances are unequal between the groups).
The default is to test for increasing variances.

order.by

either a vector z or a formula with a single explanatory variable like ~ z. The
observations in the model are ordered by the size of z. If set to NULL (the default)
the observations are assumed to be ordered.

sig.level

significance level. Default value: sig.level = 0.05.

details

logical value indicating whether specific details about the test should be re-
turned.

hyp

logical value indicating whether the Hypotheses should be returned.

Value

A list object including:

hyp  character matrix of hypotheses (if hyp = TRUE).
results a data frame of basic test results.
hreg1 matrix of regression results in Group I.
stats1 additional statistic of regression in Group I.
hreg2 matrix of regression results in Group II.
stats2 additional statistic of regression in Group II.
nulldist type of the Null distribution with its parameters.

References

Statistical Association 60, 539-547.

See Also

wh.test, gqtest.

Examples

## 5 observations in group 1 with the hypothesis that the variance of group 2 is larger
gq.test(rent ~ dist, split = 5, ah = "increasing", data = data.rent)

## Ordered by population size
eu.mod <- ols(expend ~ pop + gdp + farm + votes + mship, data = data.eu)
results <- gq.test(eu.mod, split = 13, order.by = data.eu$pop, details = TRUE)
results
plot(results)
hcc  

**Heteroskedasticity Corrected Covariance Matrix**

Description

Calculates Whites (1980) heteroskedasticity corrected covariance matrix in a linear model.

Usage

```r
hcc(mod, data = list(), digits = 4)
```

Arguments

- `mod`: estimated linear model object or formula.
- `data`: if `mod` is a formula then the corresponding data frame has to be specified.
- `digits`: number of decimal digits in rounded values.

Value

The heteroskedasticity corrected covariance matrix.

References


See Also

- `wh.test`, `bptest`.

Examples

```r
rent.est <- ols(rent ~ dist, data = data.rent)
hcc(rent.est)

hcc(wage ~ educ + age, data = data.wage)
```
Estimating Linear Models under AR(1) Autocorrelation with Hildreth and Lu Method

Description

If autocorrelated errors can be modeled by an AR(1) process (rho as parameter) then this function finds the rho value that that minimizes SSR in a Prais-Winsten transformed linear model. This is known as Hildreth and Lu estimation. The object returned by this command can be plotted using the `plot()` function.

Usage

hilu(mod, data = list(), range = seq(-1, 1, 0.01), details = FALSE)

Arguments

mod estimated linear model object or formula.
data data frame to be specified if mod is a formula.
range defines the range and step size of rho values.
details logical value, indicating whether details should be printed.

Value

A list object including:

- `results` data frame of basic regression results.
- `idx.opt` index of regression that minimizes SSR.
- `nregs` number of regressions performed.
- `rho.opt` rho-value of regression that minimizes SSR.
- `y.trans` optimal transformed y-values.
- `X.trans` optimal transformed x-values (incl. z).
- `all.regs` data frame of regression results for all considered rho values.
- `rho.vals` vector of used rho values.

References


Examples

sales.est <- ols(sales ~ price, data = data.filter)

## In this example regressions over 199 rho values between -1 and 1 are carried out
## The one with minimal SSR is printed out
ivr(sales.est)

## Direct usage of a model formula
X <- hlu(sick ~ jobless, data = data.sick[1:14,], details = TRUE)

## Print full details
X

## Suppress details
print(X, details = FALSE)

## Plot SSR over rho-values to see minimum
plot(X)

### Description

**Two-Stage Least Squares (2SLS) Instrumental Variable Regression**

**Description**

Performs a two-stage least squares regression on a single equation including endogenous regressors Y and exogenous regressors X on the right hand-side. Note that by specifying the set of endogenous regressors Y by `endog` the set of remaining regressors X are assumed to be exogenous and therefore automatically considered as part of the instrument in the first stage of the 2SLS. These variables are not to be specified in the `iv` argument. Here only instrumental variables outside the equation under consideration are specified.

**Usage**

`ivr(formula, data = list(), endog, iv, contrasts = NULL, details = FALSE, ...)`

**Arguments**

- **formula**: model formula.
- **data**: name of the data frame used. To be specified if variables are not stored in environment.
- **endog**: character vector of endogenous (to be instrumented) regressors.
- **iv**: character vector of predetermined/exogenous instrumental variables NOT already included in the model formula.
- **contrasts**: an optional list. See the contrasts.arg of `model.matrix.default`.
- **details**: logical value indicating whether details should be printed out by default.
- **...**: further arguments that `lm.fit()` understands.

**Value**

A list object including:
### Functions

- `adj.r.squ`: adjusted coefficient of determination (adj. R-squared).
- `coefficients`: IV-estimators of model parameters.
- `data/model`: matrix of the variables’ data used.
- `data.name`: name of the data frame used.
- `df`: degrees of freedom in the model (number of observations minus rank).
- `exogenous`: exogenous regressors.
- `f.hausman`: exogeneity test: F-value for simultaneous significance of all instrument parameters. If H0: "Instruments are exogenous" is rejected, usage of IV-regression can be justified against OLS.
- `f.instr`: weak instrument test: F-value for significance of instrument parameter in first stage of 2SLS regression. If H0: "Instrument is weak" is rejected, instruments are usually considered sufficiently strong.
- `fitted.values`: fitted values of the IV-regression.
- `fsd`: first stage diagnostics (weakness of instruments).
- `has.const`: logical value indicating whether model has a constant (internal purposes).
- `instrumented`: name of instrumented regressors.
- `instruments`: name of instruments.
- `model.matrix`: the model (design) matrix.
- `ncoef`: integer, giving the rank of the model (number of coefficients estimated).
- `nobs`: number of observations.
- `p.hausman`: according p-value of exogeneity test.
- `p.instr`: according p-value of weak instruments test.
- `p.values`: vector of p-values of single parameter significance tests.
- `r.squ`: coefficient of determination (R-squared).
- `residuals`: residuals in the IV-regression.
- `response`: the endogenous (response) variable.
- `shea`: Shea’s partial R-squared quantifying the ability to explain the endogenous regressors.
- `sig.squ`: estimated error variance (sigma-squared).
- `ssr`: sum of squared residuals.
- `std.err`: vector of standard errors of the parameter estimators.
- `t.values`: vector of t-values of single parameter significance tests.
- `ucov`: the (unscaled) variance-covariance matrix of the model’s estimators.
- `vcov`: the (scaled) variance-covariance matrix of the model’s estimators.
- `modform`: the model’s regression R-formula.

### References


### Examples

```r
# Numerical Illustration 20.1 in Auer (2023)
ivr(contr ~ score, endog = "score", iv = "contrprev", data = data.insurance, details = TRUE)

# Replicating an example of Ani Katchova (econometric academy)
# (https://www.youtube.com/watch?v=lm3UvcDa2Hc)
# on U.S. Women’s Labor-Force Participation (data from Wooldridge 2013)
library(wooldridge)
data(mroz)
```
# Select only working women
mroz = mroz[mroz"inlf" == 1,]
mroz = mroz[, c("lwage", "educ", "exper", "expersq", "fatheduc", "motheduc")]
attach(mroz)

# Regular ols of lwage on educ, where educ is suspected to be endogenous
# hence estimators are biased
ols(lwage ~ educ, data = mroz)

# Manual calculation of ols coeff
Sxy(educ, lwage)/Sxy(educ)

# Manual calculation of iv regression coeff
# with fatheduc as instrument for educ
Sxy(fatheduc, lwage)/Sxy(fatheduc, educ)

# Calculation with 2SLS
educ_hat = ols(educ ~ fatheduc)$fitted
ols(lwage ~ educ_hat)

# Verify that educ_hat is completely determined by values of fatheduc
head(cbind(educ,fatheduc,educ_hat), 10)

# Calculation with ivr()
ivr(lwage ~ educ, endog = "educ", iv = "fatheduc", data = mroz, details = TRUE)

# Multiple regression model with 1 endogenous regressor (educ)
# and two exogenous regressors (exper, expersq)

# Biased ols estimation
ols(lwage ~ educ + exper + expersq, data = mroz)

# Unbiased 2SLS estimation with fatheduc and motheduc as instruments
# for the endogenous regressor educ
ivr(lwage ~ educ + exper + expersq,
   endog = "educ", iv = c("fatheduc", "motheduc"),
   data = mroz)

# Manual 2SLS
# First stage: Regress endog. regressor on all exogen. regressors
# and instruments -> get exogenous part of educ
stage1.mod = ols(educ ~ exper + expersq + fatheduc + motheduc)
educ_hat = stage1.mod$fitted

# Second stage: Replace endog regressor with predicted value educ_hat
# See the uncorrected standard errors!
stage2.mod = ols(lwage ~ educ_hat + exper + expersq, data = mroz)

## Simple test for endogeneity of educ:
## Include endogenous part of educ into model and see if it is signif.
## (is signif. at 10% level)
what = ols(educ ~ exper + expersq + fatheduc + motheduc)$resid
ols(lwage ~ educ + exper + expersq + uhat)
detach(mroz)

---

**jb.test**  

**Jarque-Bera Test**

### Description

Jarque-Bera test for normality. The object of test results returned by this command can be plotted using the `plot()` function.

### Usage

```r
jb.test(x, data = list(), sig.level = 0.05, details = FALSE, hyp = TRUE)
```

### Arguments

- `x` a numeric vector, an estimated linear model object or model formula (with `data` specified). In the two latter cases the model's residuals are tested for normality.
- `data` if `mod` is a formula then the corresponding data frame has to be specified.
- `sig.level` significance level. Default value: `sig.level = 0.05`.
- `details` logical value indicating whether specific details about the test should be returned.
- `hyp` logical value indicating whether the hypotheses should be returned.

### Details

Under H0 the test statistic of the Jarque-Bera test follows a chi-squared distribution with 2 degrees of freedom. If moment of order 3 (skewness) differs significantly from 0 and/or moment of order 4 (kurtosis) differs significantly from 3, H0 is rejected.

### Value

A list object including:

- `hyp` character matrix of hypotheses (if `hyp` = TRUE).
- `results` a data frame of basic test results.
- `skew` moment of order 3 (asymmetry, skewness).
- `kur` moment of order 4 (kurtosis).
- `nobs` number of observations (internal purpose).
- `nulldist` type of the Null distribution and its parameter(s).
References


See Also

‘jarque.test()’ in Package ‘moments’.

Examples

## Test response variable for normality
X <- jb.test(data.income$loginc)
X

## Estimate linear model
income.est <- ols(loginc ~ logsave + logsum, data = data.income)
## Test residuals for normality, print details
jb.test(income.est, details = TRUE)

## Equivalent test
jb.test(loginc ~ logsave + logsum, data = data.income, details = TRUE)

## Plot the test result
plot(X)

---

**lagk**

1 to k-Period Lags of Given Vector

Description

Generates a matrix of a given vector and its 1 to k-period lags. Missing values due to lag are filled with NAs.

Usage

`lagk(u, lag = 1, delete = TRUE)`

Arguments

- `u`: a vector of one variable, usually residuals.
- `lag`: the number of periods up to which lags should be generated.
- `delete`: logical value indicating whether missing data should be eliminated from the resulting matrix.

Value

Matrix of vector u and its 1 to k-period lags.
Examples

```r
u = round(rnorm(10),2)
lagk(u)
lagk(u, lag = 3)
lagk(u, lag = 3, delete = FALSE)
```

Description

This command generates a data frame of two variables, x and y, which can be both transformed by a normalized, lambda-deformed logarithm (aka. Box-Cox-transformation). The purpose of this command is to generate data sets that represent a non-linear relationship between exogenous and endogenous variable. These data sets can be used to train linearization and heteroskedasticity issues. Note that the error term is also transformed to make it normal and homoscedastic after re-transformation to linearity. This is why generated data sets may have non-constant variance depending on the transformation parameters.

Usage

```r
makedata.bc(
  lambda.x = 1,
  lambda.y = 1,
  a = 0,
  x.max = 5,
  n = 200,
  sigma = 1,
  seed = NULL
)
```

Arguments

- `lambda.x`: deformation parameter for the x-values: -1 = inverse, 0 = log, 0.5 = root, 1 = linear, 2 = square ...
- `lambda.y`: deformation parameter for the y-values (see `lambda.x`).
- `a`: additive constant to shift the data in vertical direction.
- `x.max`: upper border of x values, must be greater than 1.
- `n`: number of artificial observations.
- `sigma`: standard deviation of the error term.
- `seed`: randomization seed.

Value

Data frame of x- and y-values.
Examples
## Compare 4 data sets generated differently
parOrg = par("mfrow")
par(mfrow = c(2,2))

## Linear data shifted by 3
A.dat <- makedata.bc(a = 3)

## Log transformed y-data
B.dat <- makedata.bc(lambda.y = 0, n = 100, sigma = 0.2, x.max = 2, seed = 123)

## Concave scatter
C.dat <- makedata.bc(lambda.y = 6, sigma = 0.4, seed = 12)

## Concave scatter, x transf.
D.dat <- makedata.bc(lambda.x = 0, lambda.y = 6, sigma = 0.4, seed = 12)

plot(A.dat, main = "linear data shifted by 3")
plot(B.dat, main = "log transformed y-data")
plot(C.dat, main = "concave scatter")
plot(D.dat, main = "concave scatter, x transf.")
par(mfrow = parOrg)

makedata.corr

Generate Exogenous Normal Data with Specified Correlations

Description
This command generates a data frame of exogenous normal regression data with given correlation between the variables. This can, for example, be used for analyzing the effects of autocorrelation.

Usage
makedata.corr(n = 10, k = 2, CORR, sample = FALSE)

Arguments
n number of observations to be generated.
k number of exogenous variables to be generated.
CORR (k x k) Correlation matrix that specifies the desired correlation structure of the data to be generated. If not specified a random positive definite covariance matrix will be used.
sample logical value indicating whether the correlation structure is applied to the population (false) or the sample (true).

Value
The generated data frame of exogenous variables.
Examples

```r
## Generate desired correlation structure
corr.mat <- cbind(c(1, 0.7), c(0.7, 1))

## Generate 10 observations of 2 exogenous variables
X <- makedata.corr(n = 10, k = 2, CORR = corr.mat)
cor(X) # not exact values of corr.mat

## Same structure applied to a sample
X <- makedata.corr(n = 10, k = 2, CORR = corr.mat, sample = TRUE)
cor(X) # exact values of corr.mat
```

mc.table

Generate \( R^2 \) Matrix of all Possible Regressions Among Regressors to Check Multicollinearity

Description

For a given set of regressors this command calculates the coefficient of determination of a regression of one specific regressor on all combinations of the remaining regressors. This provides an overview of potential multicollinearity. Needs at least three variables. For just two regressors the square of \( \text{cor}() \) can be used.

Usage

```r
mc.table(x, intercept = TRUE, digits = 3)
```

Arguments

- `x`: data frame of variables to be regressed on each other.
- `intercept`: logical value specifying whether regression should have an intercept.
- `digits`: number of digits to be rounded to.

Value

Matrix of R-squared values. The column headers indicate the respective endogenous variables that is projected on a combination of exogenous variables. Example: If we have 4 regressors \( x_1, x_2, x_3, x_4 \), then the first column of the returned matrix has 7 rows including the R-squared values of the following regressions:

1. \( x_1 \sim x_2 + x_3 + x_4 \)
2. \( x_1 \sim x_3 + x_4 \)
3. \( x_1 \sim x_2 + x_4 \)
4. \( x_1 \sim x_2 + x_3 \)
5. \( x_1 \sim x_4 \)
6. \( x_1 \sim x_3 \)
7. \( x_1 \sim x_2 \)

The second column corresponds to the regressions:

1. \( x_2 \sim x_1 + x_3 + x_4 \)
2. \( x_2 \sim x_3 + x_4 \)
3. \( x_2 \sim x_1 + x_4 \)
4. \( x_2 \sim x_1 + x_3 \)
5. \( x_2 \sim x_4 \)
6. \( x_2 \sim x_3 \)
7. \( x_2 \sim x_1 \)

and so on.

Examples

```r
## Replicate table 21.3 in the textbook
mc.table(data.printer[, -1])
```

<table>
<thead>
<tr>
<th>new.session</th>
<th>R Session Reset</th>
</tr>
</thead>
</table>

Description

`new.session` removes all objects from global environment, removes all plots, clears the console, and restores parameter settings. As default, sets the working directory to source file location in case the function is used from an R script. As an option, resets the scientific notation (e.g., 1e-04).

Usage

`new.session(cd = TRUE, sci = FALSE)`

Arguments

- `cd` if `cd = FALSE`, the working directory is not be changed. The default, `cd = TRUE`, sets the working directory to source file location.
- `sci` if `sci = TRUE`, the scientific notation is reset to the R standard option.

Value

None.

Examples

# No example available to avoid possibly unwanted object deletion in user environment.
ols

 Ordinary Least Squares Regression

Description

Estimates linear models using ordinary least squares estimation. Generated objects should be compatible with commands expecting objects generated by `lm()`. The object returned by this command can be plotted using the `plot()` function.

Usage

```r
ols(
  formula,
  data = list(),
  na.action = NULL,
  contrasts = NULL,
  details = FALSE,
  ...)
```

Arguments

- `formula` model formula.
- `data` name of data frame of variables in `formula`.
- `na.action` function which indicates what should happen when the data contain NAs.
- `contrasts` an optional list. See the `contrasts.arg` of `model.matrix.default`.
- `details` logical value indicating whether details should be printed out by default.
- `...` other arguments that `lm.fit()` supports.

Details

Let X be a model object generated by `ols()` then `plot(X, ...)` accepts the following arguments:

- `pred.int = FALSE` should prediction intervals be added to plot?
- `conf.int = FALSE` should confidence intervals be added to plot?
- `residuals = FALSE` should residuals be added to plot?
- `center = FALSE` should mean values of both variables be added to plot?

Value

A list object including:

- `coefficients/coef` estimated parameters of the model.
- `residuals/resid` residuals of the estimation.
- `effects` n vector of orthogonal single-df effects. The first rank of them correspond to non-aliased coefficients, a
fitted.values
df.residual/df
degrees of freedom in the model (number of observations minus rank).
se
vector of standard errors of the parameter estimators.
t.value
vector of t-values of single parameter significance tests.
p.value
vector of p-values of single parameter significance tests.
data/model
matrix of the variables’ data used.
response
the endogenous (response) variable.
model.matrix
the model (design) matrix.
se
vector of standard errors of the parameter estimators.
t.value
vector of t-values of single parameter significance tests.
p.value
vector of p-values of single parameter significance tests.
data/model
matrix of the variables’ data used.
response
the endogenous (response) variable.
model.matrix
the model (design) matrix.
se
vector of standard errors of the parameter estimators.
t.value
vector of t-values of single parameter significance tests.
p.value
vector of p-values of single parameter significance tests.
data/model
matrix of the variables’ data used.
response
the endogenous (response) variable.
model.matrix
the model (design) matrix.

Examples

## Minimal simple regression model
check <- c(10,30,50)
tip <- c(2,3,7)
tip.est <- ols(tip ~ check)

## Equivalent estimation using data argument
tip.est <- ols(y ~ x, data = data.tip)

## Show estimation results
tip.est

## Show details
print(tip.est, details = TRUE)

## Plot scatter and regression line
plot(tip.est)

## Plot confidence (dark) and prediction bands (light), residuals and two center lines
plot(tip.est, pred.int = TRUE, conf.int = TRUE, residuals = TRUE, center = TRUE)

## Multiple regression model
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer), details = TRUE)
fert.est
### ols.has.const  
*
**Check if Model has a Constant**

**Description**

Checks if a linear model included a constant level parameter (alpha).

**Usage**

```r
ols.has.const(mod)
```

**Arguments**

- `mod` linear model object of class "desk" or "lm".

**Value**

A logical value: TRUE (has constant) or FALSE (has no constant).

**Examples**

```r
my.modA = ols(y ~ x, data = data.tip)
my.modB = ols(y ~ 0 + x, data = data.tip)
ols.has.const(my.modA)
ols.has.const(my.modB)
```

---

### ols.infocrit  
*
**Calculate Common Information Criteria**

**Description**

Calculates three common information criteria of models estimated by `ols()`.

**Usage**

```r
ols.infocrit(mod, which = "all", scaled = FALSE)
```

**Arguments**

- `mod` linear model object generated by `ols()`.
- `which` string value specifying the type of criterion: "aic" (Akaike Information Criterion), "sic" (Schwarz Information Criterion), or "pc", (Prognostic Criterion), optional, if omitted then all criteria are returned ("all").
- `scaled` logical value which indicates whether criteria should be scaled by the number of observations T.
Value

A data frame of AIC, SIC, and PC values.

Examples

```r
wage.est <- ols(wage ~ educ + age, data = data.wage)
ols.infocrit(wage.est) # Return all criteria unscaled
ols.infocrit(wage.est, scaled = TRUE) # Return all criteria scaled
ols.infocrit(wage.est, which = "pc") # Return Prognostic Criterion unscaled
```

---

**ols.interval**

*Calculate Different Types of Intervals in a Linear Model*

**Description**

Calculates different types of intervals in a linear model.

**Usage**

```r
ols.interval(
  mod, data = list(),
  type = c("confidence", "prediction", "acceptance"),
  which.coef = "all",
  sig.level = 0.05,
  q = 0,
  dir = c("both", "left", "right"),
  xnew,
  details = FALSE
)
```

**Arguments**

- `mod` linear model object generated by `ols()`.
- `data` name of data frame to be specified if `mod` is a formula.
- `type` string value indicating the type of interval to be calculated. Default is "confidence".
- `which.coef` strings of variable name(s) or vector of indices indicating the coefficients in the linear model for which confidence or acceptance intervals should be calculated. By default all coefficients are selected. Ignored for prediction intervals.
- `sig.level` significance level.
- `q` value against which null hypothesis is tested. Only to be specified if `type` = "acceptance".
- `dir` direction of the alternative hypothesis underlying the acceptance intervals. One sided confidence- and prediction intervals are not (yet) supported.
xnew

(T \times K) matrix of new values of the exogenous variables, at which interval should be calculated, where T is the number of exogenous data points at which intervals should be calculated K is the number of exogenous variables in the model If type = "prediction" then prediction intervals are calculated at xnew, if type = "confidence" then confidence intervals around the unknown true y-values are calculated at xnew (aka. confidence band). Ignored if type = "acceptance". In multiple regression models variable names must be specified.

details

logical value indicating whether details (estimated standard deviations) should be printed out.

Value

A list object including:

- **results**
  interval borders (lower and upper) and center of interval (if dir = "both").
- **std.err**
  estimated standard deviations.
- **t.value**
  critical t-value.

Examples

```r
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer))
my.mat = cbind(x1 = log(c(6,3,9)), x2 = log(c(5,3,10)))

## 95% CI for all parameters
ols.interval(fert.est)

## 95% CI for intercept and beta2
ols.interval(fert.est, which.coef = c(1,3))

## 95% CI around three true, constant y-values
ols.interval(fert.est, xnew = my.mat)

## AI for H0:beta1 = 0.5 and H0:beta2 = 0.5
ols.interval(fert.est, type = "acc", which.coef = c(2,3), q = 0.5)

## AI for H0:beta1 <= 0.5
ols.interval(fert.est, type = "acc", which.coef = 2, dir = "right", q = 0.5)

## PI (Textbook p. 285)
ols.interval(fert.est, type = "pred", xnew = c(x1 = log(29), x2 = log(120)), details = TRUE)

## Three PI
ols.interval(fert.est, type = "pred", xnew = my.mat, details = TRUE)
```
Description

Calculates the predicted values of a linear model based on specified values of the exogenous variables. Optionally the estimated variance of the prediction error is returned.

Usage

```r
ols.predict(mod, data = list(), xnew, antilog = FALSE, details = FALSE)
```

Arguments

- `mod`: model object generated by `ols()` or `lm()`.
- `data`: name of data frame to be specified if `mod` is a formula.
- `xnew`: (\(T \times K\)) matrix of new values of the exogenous variables, for which a prediction should be made, where \(K\) is the number of exogenous variables in the model \(T\) is the number of predictions to be made. If `xnew` is not specified, the fitted values are returned.
- `antilog`: logical value which indicates whether to re-transform the predicted value of a log transformed dependent variable back into original units.
- `details`: logical value, if specified as `TRUE`, a list is returned, which additionally includes the estimated variance of the prediction error (`var.pe`), estimated variance of the error term (`sig.squ`), and the estimated sampling error (`smpl.err`).

Value

A list object including:

- `pred.val`: the predicted values.
- `xnew`: values of predictor at which predictions should be evaluated.
- `var.pe`: estimated variance of prediction error.
- `sig.squ`: estimated variance of error term.
- `smpl.err`: estimated sampling error.
- `mod`: the model estimated (for internal purposes)

Examples

```r
## Estimate logarithmic model
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer))

## Set new x data
my.mat = cbind(x1 = log(c(6,3,9)), x2 = log(c(5,3,10)))

## Returns fitted values
ols.predict(fert.est)

## Returns predicted values at new x-values
ols.predict(fert.est, xnew = my.mat)

## Returns re-transformed predicted values and est. var. of pred. error
```

---

**ols.predict**

Description

Calculates the predicted values of a linear model based on specified values of the exogenous variables. Optionally the estimated variance of the prediction error is returned.

Usage

```r
ols.predict(mod, data = list(), xnew, antilog = FALSE, details = FALSE)
```

Arguments

- `mod`: model object generated by `ols()` or `lm()`.
- `data`: name of data frame to be specified if `mod` is a formula.
- `xnew`: (\(T \times K\)) matrix of new values of the exogenous variables, for which a prediction should be made, where \(K\) is the number of exogenous variables in the model \(T\) is the number of predictions to be made. If `xnew` is not specified, the fitted values are returned.
- `antilog`: logical value which indicates whether to re-transform the predicted value of a log transformed dependent variable back into original units.
- `details`: logical value, if specified as `TRUE`, a list is returned, which additionally includes the estimated variance of the prediction error (`var.pe`), estimated variance of the error term (`sig.squ`), and the estimated sampling error (`smpl.err`).

Value

A list object including:

- `pred.val`: the predicted values.
- `xnew`: values of predictor at which predictions should be evaluated.
- `var.pe`: estimated variance of prediction error.
- `sig.squ`: estimated variance of error term.
- `smpl.err`: estimated sampling error.
- `mod`: the model estimated (for internal purposes)

Examples

```r
## Estimate logarithmic model
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer))

## Set new x data
my.mat = cbind(x1 = log(c(6,3,9)), x2 = log(c(5,3,10)))

## Returns fitted values
ols.predict(fert.est)

## Returns predicted values at new x-values
ols.predict(fert.est, xnew = my.mat)

## Returns re-transformed predicted values and est. var. of pred. error
```
ols.predict(fert.est, xnew = my.mat, antilog = TRUE, details = TRUE)

par.f.test  

F-test on Multiple Linear Combinations of Estimated Parameters in a Linear Model

Description

Performs an F-test (non-directional) on multiple (L) linear combinations of parameters in a linear model.

Usage

par.f.test(
  mod,  
data = list(),  
  nh,  
  q = rep(0, dim(nh)[1]),  
  sig.level = 0.05,  
  details = FALSE,  
  hyp = TRUE
)

Arguments

mod  
model object estimated by ols() or lm().

data  
name of the data frame to be used if mod is only a formula.

nh  
matrix of the coefficients of the linear combination of parameters. Each of the L rows of that matrix represents a linear combination.

q  
L-dimensional vector of values on which the parameter (combination) is to be tested against. Default value is the null-vector.

sig.level  
significance level. Default value: sig.level = 0.05.

details  
logical value indicating whether specific details about the test should be returned.

hyp  
logical value indicating whether the hypotheses should be part of the output. To be disabled if output is too large.

Details

Objects x generated by par.f.test can be plotted using plot(x, plot.what = ...). Argument plot.what can have the following values:

"dist"  
plot the null distribution, test statistics and p-values.

"ellipse"  
plot acceptance ellipse.
If `plot.what = "ellipse"` is specified, further arguments can be passed to `plot()`:

- `type = "acceptance"`: plot acceptance ellipse ("acceptance") or confidence ellipse ("confidence").
- `which.coef = c(2,3)`: for which two coefficients should the ellipse be plotted?
- `center = TRUE`: plot center of ellipse.
- `intervals = TRUE`: plot interval borders.
- `test.point = TRUE`: plot the point (q-values or coefficients) used in F-Test.
- `q = c(0,0)`: the q-value used in acceptance ellipse.
- `sig.level = 0.05`: significance level used.

### Value

A list object including:

- `hyp`: character matrix of hypotheses (if `hyp = TRUE`).
- `nh`: linear combinations tested in the null hypothesis (in matrix form).
- `q`: vector of values the linear combinations are tested on.
- `mod`: the model passed to `par.f.test`.
- `results`: a data frame of basic test results.
- `SSR.H0`: sum of squared residuals in H0-model.
- `SSR.H1`: sum of squared residuals in regular model.
- `nulldist`: type of the null distribution with its parameters.

### Examples

```r
## H0: beta1 = 0.33 and beta2 = 0
x <- par.f.test(barley ~ phos + nit, data = log(data.fertilizer),
              nh = rbind(c(0,1,0), c(0,0,1)),
              q = c(0.33,0.33),
              details = TRUE)

x # Show the test results

plot(x) # Visualize the test result
plot(x, plot.what = "ellipse", q = c(0.33, 0.33))
```

---

**par.t.test**

**t-Test on Estimated Parameters of a Linear Model**

**Description**

Performs a t-test on a single parameter hypothesis or a hypothesis containing a linear combination of parameters of a linear model. The object of test results returned by this command can be plotted using the `plot()` function.
par.t.test

Usage

par.t.test(
  mod,
  data = list(),
  nh,
  q = 0,
  dir = c("both", "left", "right"),
  sig.level = 0.05,
  details = FALSE,
  hyp = TRUE
)

Arguments

mod          model object estimated by ols() or lm().
data         name of the data frame to be used if mod is a formula and the variables are not present in the environment.

Arguments

nh          vector of the coefficients of the linear combination of parameters.
q            value on which parameter (combination) is to be tested against. Default value: q = 0.
dir          direction of the hypothesis: "both", "left", "right", Default value: "both".
sig.level          significance level. Default value: sig.level = 0.05.
details          logical value indicating whether specific details about the test should be returned.
hyp          logical value indicating whether the Hypotheses should be returned.

Value

A list object including:

hyp          character matrix of hypotheses (if hyp = TRUE).

Examples

## Test H1: "phos + nit <> 1"
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer))
x = par.t.test(fert.est, nh = c(0,1,1), q = 1, details = TRUE)
x # Show the test results

plot(x) # Visualize the test result
## Test H1: "phos > 0.5"

```r
x = par.t.test(fert.est, nh = c(0,1,0), q = 0.5, dir = "right")
plot(x)
```

---

### pc.test

**Prognostic Chow Test on Structural Break**

**Description**

Performs prognostic Chow test on structural break. The object of test results returned by this command can be plotted using the `plot()` function.

**Usage**

```r
pc.test(
  mod,
  data = list(),
  split,
  sig.level = 0.05,
  details = FALSE,
  hyp = TRUE
)
```

**Arguments**

- `mod` the regular model (estimated or formula) without dummy variables.
- `data` if `mod` is a formula then the corresponding data frame has to be specified.
- `split` number of periods in phase I (last period before suspected break). Phase II is the total of remaining periods.
- `sig.level` significance level. Default value: `sig.level = 0.05`.
- `details` logical value indicating whether specific details (null distribution, number of periods, and SSRs) of the test should be displayed.
- `hyp` logical value indicating whether the hypotheses should be displayed.

**Value**

A list object including:

- `hyp` the null-hypothesis to be tested.
- `results` data frame of test results.
- `SSR1` sum of squared residuals of phase I.
- `SSR` sum of squared residuals of phase I + II.
- `periods1` number of periods in Phase I.
- `periods.total` total number of periods.
- `nulldist` the null distribution in the test.
References

Examples

```r
## Estimate model
unemp.est <- ols(unempl ~ gdp, data = data.unempl[1:14,])

## Test for immediate structural break after t = 13
X <- pc.test(unemp.est, split = 13, details = TRUE)
X
plot(X)
```

---

**pdw**

*Durbin-Watson Distribution*

**Description**

**Usage**

```r
pdw(x, mod, data = list())
```

**Arguments**

- `x`: quantile value(s) at which the density should be determined.
- `mod`: estimated linear model object, formula (with data specified), or model matrix.
- `data`: if `mod` is a formula then the name of the corresponding data frame has to be specified.

**Details**
Distribution depends on values of the exogenous variables. That is why it must be calculated from each specific data set, respectively.

**Value**
Numerical density value(s).

**References**
See Also
ddw, dw.test.

Examples

```r
filter.est <- ols(sales ~ price, data = data.filter)
pdw(x = c(0.9, 1.7, 2.15), filter.est)
```

Description

This function implements an S3 method for plotting regression- and test-results generated by functions of the desk package. Used for internal purposes.

Usage

```r
## S3 method for class 'desk'
plot(x, ...)
```

Arguments

- `x`: object of class desk to be plotted.
- `...`: any argument that `plot()` accepts.

Value

No return value. Called for side effects.

Examples

```r
## Test H1: "phos + nit <> 1"
fert.est <- ols(barley ~ phos + nit, data = log(data.fertilizer))
x = par.t.test(fert.est, nh = c(0,1,1), q = 1, details = TRUE)
x # Show the test results
class(x) # Check its class
plot(x) # Visualize the test result

## Plot confidence (dark) and prediction bands (light), residuals and two center lines
## in a simple regression model
tip.est <- ols(y ~ x, data = data.tip)
class(x) # Check its class
plot(tip.est, pred.int = TRUE, conf.int = TRUE, residuals = TRUE, center = TRUE)
```
print.desk

Alternative Console Output for Regression- and Test-results

Description

This function implements an S3 method for printing regression- and test-results generated by functions of the desk package. Used for internal purposes.

Usage

```r
## S3 method for class 'desk'
print(x, details, digits = 4, ...)
```

Arguments

- **x**: object of class desk to be printed to the console.
- **details**: logical value indicating whether details of object x should be printed.
- **digits**: number of digits to round to (only output).
- **...**: any argument that `print()` accepts.

Value

No return value. Called for side effects.

Examples

```r
## Simple regression model
tip.est <- ols(y ~ x, data = data.tip)

## Check its class
class(tip.est)
#> [1] "desk" "lm"

## Standard regression output
print(tip.est) # same as tip.est

## Regression output with details rounded to 2 digits
print(tip.est, details = TRUE, digits = 2)
```
**qlr.cv**  
*Calculates the critical value in a Quandt Likelihood Ratio-Test for Structural Breaks in a Parameter with Unknown Break Date*

**Description**

Calculates critical values for Quandt Likelihood Ratio-test (QLR) for structural breaks with unknown break date.

**Usage**

```r
qlr.cv(tAll, from = round(0.15*tAll), to = round(0.85*tAll),
       L = 2, sig.level = list(0.05, 0.01, 0.1))
```

**Arguments**

- `tAll`: sample size.
- `from`: start period of range to be analyzed for a break.
- `to`: end period of range to be analyzed for a break.
- `L`: number of parameters.
- `sig.level`: significance level. Allowed values are 0.01, 0.05 or 0.10.

**Value**

A list object including:

- `lambda`: the lambda correction value for the critical value.
- `range`: range of values.
- `cv.chi2`: critical value of chi^2-test statistics.
- `cv.f`: critical value of F-test statistics.

**References**


**Examples**

```r
qlr.cv(20, L = 2, sig.level = 0.01)
```
qlr.test  

Quandt Likelihood Ratio-Test for Structural Breaks in any Parameter with Unknown Break Date

Description

Performs Quandt Likelihood Ratio-test (QLR) for structural breaks with unknown break date. The object returned by this command can be plotted using the plot() function.

Usage

qlr.test(mod, data = list(), from, to, sig.level = 0.05, details = FALSE)

Arguments

mod  
the regular model object (without dummies) estimated by ols() or lm().

data  
name of the data frame to be used if mod is only a formula.

from  
start period of range to be analyzed for a break.

to  
end period of range to be analyzed for a break.

sig.level  
significance level. Allowed values are 0.01, 0.05 or 0.10.

details  
logical value indicating whether specific details about the test should be returned.

Value

A list object including:

hyp  
the null-hypothesis to be tested.

results  
data frame of test results.

chi2.stats  
χ²-test statistics calculated between from and to.

f.stats  
F-test statistics calculated between from and to.

f.crit  
lower and upper critical F-value.

p.value  
p-value in the test using approximation method proposed by Hansen (1997).

breakpoint  
period at which largest F-value occurs.

periods  
the range of periods analyzed.

lf.crit  
lower and upper critical F-value including corresponding lambda values.

lambda  
the lambda correction value for the critical value.

References

**Examples**

```r
unemp.est <- ols(unempl ~ gdp, data = data.unempl)
my.qlr <- qlr.test(unemp.est, from = 13, to = 17, details = TRUE)
my.qlr # Print test results

plot(my.qlr) # Plot test results
```

---

**repeat.sample**

*Generates OLS Data and Confidence/Prediction Intervals for Repeated Samples*

---

**Description**

This command simulates repeated samples given fixed data of the exogenous predictors and given (true) regression parameters. For each sample generated the results from an OLS regression with level parameter and confidence intervals (CIs) as well as prediction intervals are calculated.

**Usage**

```r
repeat.sample(
  x, 
  true.par, 
  omit = 0, 
  mean = 0, 
  sd = 1, 
  rep = 100, 
  xnew = x, 
  sig.level = 0.05, 
  seed = NULL 
)
```

**Arguments**

- `x` (n x k) vector or matrix of exogenous data, where each column represents the data of one of k exogenous predictors. The number of rows represents the sample size n.
- `true.par` vector of true parameters in the linear model (level and slope parameters). If `true.par` is a vector without named elements then coefficients are named "alpha", "beta1", "beta2", ..., "betak" by default. Otherwise the names specified are used.
- `omit` vector of indices identifying the exogenous variables to be omitted in the true model, e.g. `omit = 1` corresponds to the first exogenous variable to be omitted. This argument can be used to illustrate omitted variable bias in parameter and standard error estimates. Default value is `omit = 0`, i.e. no exogenous variable is omitted.
mean expected value of the normal distribution of the error term.

sd standard deviation of the normal distribution of the error term. Used only for generating simulated y-values. Interval estimators use the estimated sigma.

rep repetitions, i.e. number of simulated samples. The samples in each matrix generated have enumerated names "SMPL1", "SMPL2", ..., "SMPLs".

xnew \((t \times k)\) matrix of new exogenous data points at which prediction intervals should be calculated. \(t\) corresponds to the number of new data points, \(k\) to the number of exogenous variables in the model. If not specified regular values \(x\) are used (see first argument).

sig.level significance level for confidence and prediction intervals.

seed optionally set random seed to arbitrary number if results should be made replicable.

Details

Let \(X\) be an object generated by \texttt{repeat.sample()} then \texttt{plot(X, ...)} accepts the following arguments:

\begin{itemize}
  \item \texttt{plot.what = "confint"} plot stacked confidence intervals for all samples. Additional arguments are \texttt{center = TRUE} (plot center of intervals?), \texttt{which.coef = 2} (intervals for which coefficient?), \texttt{center.size = 1} (size of the center dot), \texttt{lwd = 1} (line width).
  \item \texttt{plot.what = "reglines"} plot regression lines of all samples.
  \item \texttt{plot.what = "scatter"} plot scatter plots of all samples.
\end{itemize}

Value

A list of named data structures. Let \(s = \) number of samples, \(n = \) sample size, \(k = \) number of coefficients, \(t = \) number of new data points in \(x_{\text{new}}\) then:

\begin{itemize}
  \item \texttt{x} \((n \times k\) matrix): copy of data of exogenous regressors that was passed to the function.
  \item \texttt{y} \((n \times s\) matrix): simulated real y values in each sample.
  \item \texttt{fitted} \((n \times s\) matrix): estimated y values in each sample.
  \item \texttt{coef} \((k \times s\) matrix): estimated parameters in each sample.
  \item \texttt{true.par} \((k\) vector): vector of true parameter values (implemented only for \texttt{plot.confint()}).
  \item \texttt{u} \((n \times s\) matrix): random error term in each sample.
  \item \texttt{residuals} \((n \times s\) matrix): residuals of OLS estimations in each sample.
  \item \texttt{sig.squ} \((s\) vector): estimated variance of the error term in each sample.
  \item \texttt{var.u} \((s\) vector): variance of random errors drawn in each sample.
  \item \texttt{se} \((k \times s\) matrix): estimated standard deviation of the coefficients in each sample.
  \item \texttt{vcov.coef} \((k \times k \times s\) array): estimated variance-covariance matrix of the coefficients in each sample.
  \item \texttt{confint} \((k \times 2 \times s\) array): confidence intervals of the coefficients in each sample. Interval bounds are named "lower" and "upper".
  \item \texttt{outside.ci} \((k\) vector): percentage of confidence intervals not covering the true value for each of the regression parameters.
  \item \texttt{y0} \((t \times s\) matrix): simulated real future y values at \(x_{\text{new}}\) in each sample (real line plus real error).
  \item \texttt{y0.fitted} \((t \times s\) matrix): point prediction, i.e. estimated y values at \(x_{\text{new}}\) in each sample (regression line).
  \item \texttt{predint} \((t \times 2 \times s\) array): prediction intervals of future endogenous realizations at exogenous data points specified by \(x_{\text{new}}\).
  \item \texttt{sd.pe} \((t \times s\) matrix): estimated standard deviation of prediction errors at all exogenous data points in each sample.
  \item \texttt{outside.pi} \((t\) vector): percentage of prediction intervals not covering the true value \(y_0\) at \(x_{\text{new}}\).
  \item \texttt{bias.coef} \((k\) vector): true bias in parameter estimators if variables are omitted (argument \texttt{omit} unequal to zero).
\end{itemize}
Examples

## Generate data of two predictors
x1 = c(1,2,3,4,5)
x2 = c(2,4,5,5,6)
x = cbind(x1,x2)

## Generate list of data structures and name it "out"
out = repeat.sample(x, true.par = c(2,1,4), rep = 10)

## Extract some data
out$coef[2,8] # Extract estimated beta1 (i.e. 2nd coef) in the 8th sample
out$confint["beta1","SMPL8"] # Same as above using internal names
out$confint["beta1","upper","SMPL5"] # Extract only upper bound of CI of beta 1 from 5th sample
out$confint[,5] # Extract CIs (upper and lower bound) for all parameters from 5th sample
out$confint[,"SMPL5"] # Same as above using internal names
out$confint["beta1","SMPL5"] # Extract CI of beta 1 from 5th sample
out$u.hat[,"SMPL7"] # Extract residuals from OLS estimation of sample 7

## Generate prediction intervals at three specified points of exogenous data (xnew)
out = repeat.sample(x, true.par = c(2,1,4), rep = 10,
     xnew = cbind(x1 = c(1.5,6,7), x2 = c(1,3,5.5))
out$predint[,6] # Prediction intervals at the three data points of xnew in 6th sample
out$sd.pe[,6] # Estimated standard deviations of prediction errors in 6th sample
out$outside.pi # Percentage of how many intervals miss true y0 realization

## Illustrate that the relative shares of cases when the interval does not cover the true value approaches the significance level
out = repeat.sample(x, true.par = c(2,1,4), rep = 1000)
out$outside.ci

## Illustrate omitted variable bias
out.unbiased = repeat.sample(x, true.par = c(2,1,4))
mean(out.unbiased$coef["beta1",]) # approx. equal to beta1 = 1
out.biased = repeat.sample(x, true.par = c(2,1,4), omit = 2) # omit x2
mean(out.biased$coef["beta1",]) # not approx. equal to beta1 = 1
out.biased$bias.coef # show the true bias in coefficients

## Simulate a regression with given correlation structure in exogenous data
corr.mat = cbind(c(1, 0.9),c(0.9, 1)) # Generate desired corr. structure (high autocorrelation)
X = makedata.corr(n = 10, k = 2, CORR = corr.mat) # Generate 10 obs. of 2 exogenous variables
out = repeat.sample(X, true.par = c(2,1,4), rep = 1) # Simulate a regression
out$vcov.coef

## Illustrate confidence intervals
out = repeat.sample(c(10, 20, 30,50), true.par = c(0.2,0.13), rep = 10, seed = 12)
plot(out, plot.what = "confint")

## Illustrate normality of dependent variable
out = repeat.sample(c(10,30,50), true.par = c(0.2,0.13), rep = 200)
plot(out, plot.what = "scatter")

## Illustrate confidence bands in a regression
plot(out, plot.what = "reglines")

---

**reset.test**  
*RESET Method for Non-linear Functional Form*

**Description**

Ramsey’s RESET for non-linear functional form. The object of test results returned by this command can be plotted using the `plot()` function.

**Usage**

```r
reset.test(
  mod,
  data = list(),
  m = 2,
  sig.level = 0.05,
  details = FALSE,
  hyp = TRUE
)
```

**Arguments**

- `mod`  
  estimated linear model object or formula.
- `data`  
  if `mod` is a formula then the corresponding data frame has to be specified.
- `m`  
  the number of non-linear terms of fitted y values that should be included in the extended model. Default is `m = 2`, i.e. to add $\hat{y}^2$ and $\hat{y}^3$.
- `sig.level`  
  significance level. Default value: `sig.level = 0.05`.
- `details`  
  logical value indicating whether specific details about the test should be returned.
- `hyp`  
  logical value indicating whether the Hypotheses should be returned.

**Value**

A list object including:

- `hyp`  
  character matrix of hypotheses (if `hyp = TRUE`).
- `results`  
  a data frame of basic test results.
- `SSR0`  
  SSR of the H0-model.
- `SSR1`  
  SSR of the extended model.
- `L`  
  numbers of parameters tested in H0.
- `nulldist`  
  null distribution of the test.
References


See Also

resettest.

Examples

```r
## Numerical illustration 14.2. of the textbook
X <- resettest(milk ~ feed, m = 4, data = data.milk)
X

## Plot the test result
plot(X)
```

**rm.all**  
**Remove All Objects**

Description

Removes all objects from global environment, except those that are specified by argument `keep`.

Usage

```r
rm.all(keep = NULL)
```

Arguments

- `keep`  
a vector of strings specifying object names to be kept in environment, optional, if omitted then all objects in global environment are removed.

Value

None.

Examples

```r
# No example available to avoid possibly unwanted object deletion in user environment.
```
Description

Helps to (visually) detect whether a time series is stationary or non-stationary. A time series is a data-generating process with every observation - as a random variable - following a distribution. When expectational value, variance, and covariance (between different points in time) are constant, the time series is indicated as weekly dependent and seen as stationary. This desired property is a requirement to overcome the problem of spurious regression. Since there is no distribution but only one observation for each point in time, adjacent observations will be used as stand-in to calculate the indicators. Therefore, the chosen window should not be too large.

Usage

roll.win(x, window = 3, indicator = "mean", tau = NULL)

Arguments

x
a vector, usually a time series.

window
the width of the window to calculate the indicator.

indicator
character string specifying type of indicator: expected value ("mean"), variance ("var") or covariance ("cov").

tau
number of lags to calculate the covariance. When not specified using "cov", the variance is calculated.

Value

da vector of the calculated indicators.

Note

Objects generated by roll.win() can be plotted using the regular plot() command.

Examples

## Plot the expected values with a window of width 5
exp.values <- roll.win(1:100, window = 5, indicator = "mean")
plot(exp.values)

## Spurious regression example
set.seed(123)
N <- 10^3
p.values <- rep(NA, N)
for (i in 1:N) {
  x <- 1:100 + rnorm(100) # time series with trend
  y <- 1:100 + rnorm(100) # time series with trend
}
p.values[i] <- summary(ols(y ~ x))$coef[2,4]
}
sum(p.values < 0.05)/N  # share of significant results (100%)
for (i in 1:N) {
  x <- rnorm(100)  # time series without trend
  y <- 1:100 + rnorm(100) # time series with trend
  p.values[i] <- summary(ols(y ~ x))$coef[2,4]
}
sum(p.values < 0.05)/N  # share of significant results (~ 5%)

---

**rprofile.add**

*Add a Command to User R Startup File Rprofile.site*

**Description**

Adds a specified R command to file "Rprofile.site" for automatic execution during startup.

**Usage**

`rprofile.add(line)`

**Arguments**

- `line`  
  a text string specifying the command to be added.

**Value**

None.

**Examples**

```r
if (FALSE) rprofile.add("library(desk)")  # Makes package desk to be loaded at startup
```

---

**rprofile.open**

*Open User R Startup File Rprofile.site*

**Description**

Opens the user R startup file "Rprofile.site" for viewing or editing.

**Usage**

`rprofile.open()`
Value

None.

Examples

if (FALSE) rprofile.open() # Open the file if statement = TRUE

Sxy

Variation and Covariation

Description

Calculates the variation of one variable or the covariation of two different variables.

Usage

Sxy(x, y = x, na.rm = FALSE)

Arguments

x

vector of one variable.

y

vector of another variable (optional). If specified then the covariation of x and y is calculated. If omitted then the variation of x is calculated.

na.rm

a logical value indicating whether NA values should be stripped before the computation proceeds.

Value

The variation of x or the covariation of x and y.

Examples

x = c(1, 2)
y = c(4, 1)
Sxy(x) # variation
Sxy(x, y) # covariation

## Second example illustrating the na.rm option
x = c(1, 2, NA, 4)
Sxy(x)
Sxy(x, na.rm = TRUE)
**wh.test**

*White Heteroskedasticity Test*

**Description**

White's test for heteroskedastic errors.

**Usage**

```r
wh.test(mod, data = list(), sig.level = 0.05, details = FALSE, hyp = TRUE)
```

**Arguments**

- `mod` estimated linear model object or formula.
- `data` if `mod` is a formula then the corresponding data frame has to be specified.
- `sig.level` significance level. Default value: `sig.level = 0.05`.
- `details` logical value indicating whether specific details about the test should be returned.
- `hyp` logical value indicating whether the hypotheses should be returned.

**Value**

A list object including:

- `hyp` character matrix of hypotheses (if `hyp = TRUE`).
- `result` a data frame of basic test results.
- `hreg` matrix of aux. regression results.
- `stats` additional statistic of aux. regression.
- `nulldist` type of the null distribution with its parameters.

**References**


**See Also**

`bptest`.

**Examples**

```r
## White test for a model with two regressors
X <- wh.test(wage ~ educ + age, data = data.wage)

## Show the auxiliary regression results
X$hreg
```
## Prettier way
print(X, details = TRUE)

## Plot the test result
plot(X)
# Index

<table>
<thead>
<tr>
<th>*GlobalEnv</th>
</tr>
</thead>
<tbody>
<tr>
<td>new.session, 56</td>
</tr>
<tr>
<td>rm.all, 76</td>
</tr>
<tr>
<td>AR1 process</td>
</tr>
<tr>
<td>ar1sim, 4</td>
</tr>
<tr>
<td>hilu, 47</td>
</tr>
<tr>
<td>AR1</td>
</tr>
<tr>
<td>acc, 3</td>
</tr>
<tr>
<td>dw.test, 42</td>
</tr>
<tr>
<td>Akaike Information Criterion</td>
</tr>
<tr>
<td>ols.infocrit, 59</td>
</tr>
<tr>
<td>Box-Cox-Model</td>
</tr>
<tr>
<td>bc.model, 7</td>
</tr>
<tr>
<td>Breuch Pagan test</td>
</tr>
<tr>
<td>bp.test, 9</td>
</tr>
<tr>
<td>CI</td>
</tr>
<tr>
<td>repeat.sample, 72</td>
</tr>
<tr>
<td>Chow test</td>
</tr>
<tr>
<td>pc.test, 66</td>
</tr>
<tr>
<td>Durbin Watson test</td>
</tr>
<tr>
<td>dw.test, 42</td>
</tr>
<tr>
<td>Durbin Watson</td>
</tr>
<tr>
<td>dww, 40</td>
</tr>
<tr>
<td>Durbin-Watson</td>
</tr>
<tr>
<td>pdw, 67</td>
</tr>
<tr>
<td>F-test</td>
</tr>
<tr>
<td>par.f.test, 63</td>
</tr>
<tr>
<td>Hildreth-Lu</td>
</tr>
<tr>
<td>hilu, 47</td>
</tr>
<tr>
<td>Prognostic Criterion</td>
</tr>
<tr>
<td>ols.infocrit, 59</td>
</tr>
<tr>
<td>Quandt Likelihood Ratio test</td>
</tr>
<tr>
<td>qlr.cv, 70</td>
</tr>
<tr>
<td>qlr.test, 71</td>
</tr>
<tr>
<td>RESET</td>
</tr>
<tr>
<td>reset.test, 75</td>
</tr>
<tr>
<td>Rprofile.site</td>
</tr>
<tr>
<td>rprofile.add, 78</td>
</tr>
<tr>
<td>rprofile.open, 78</td>
</tr>
<tr>
<td>R</td>
</tr>
<tr>
<td>rprofile.add, 78</td>
</tr>
<tr>
<td>rprofile.open, 78</td>
</tr>
<tr>
<td>Schwarz Criterion</td>
</tr>
<tr>
<td>ols.infocrit, 59</td>
</tr>
<tr>
<td>VKQ</td>
</tr>
<tr>
<td>hilu, 47</td>
</tr>
<tr>
<td>White test</td>
</tr>
<tr>
<td>wh.test, 80</td>
</tr>
<tr>
<td>acceptance interval</td>
</tr>
<tr>
<td>ols.interval, 60</td>
</tr>
<tr>
<td>arguments</td>
</tr>
<tr>
<td>arguments, 6</td>
</tr>
<tr>
<td>autocorrelated errors</td>
</tr>
<tr>
<td>ar1sim, 4</td>
</tr>
<tr>
<td>autocorrelation</td>
</tr>
<tr>
<td>acc, 3</td>
</tr>
<tr>
<td>dw.test, 42</td>
</tr>
<tr>
<td>hilu, 47</td>
</tr>
<tr>
<td>autoregressive model</td>
</tr>
<tr>
<td>acc, 3</td>
</tr>
<tr>
<td>coefficient of determination</td>
</tr>
<tr>
<td>mc.table, 55</td>
</tr>
<tr>
<td>confidence band</td>
</tr>
<tr>
<td>makedata.bc, 53</td>
</tr>
<tr>
<td>confidence interval</td>
</tr>
<tr>
<td>ols.interval, 60</td>
</tr>
<tr>
<td>repeat.sample, 72</td>
</tr>
<tr>
<td>configuration</td>
</tr>
<tr>
<td>rprofile.add, 78</td>
</tr>
<tr>
<td>rprofile.open, 78</td>
</tr>
<tr>
<td>console</td>
</tr>
<tr>
<td>print.desk, 69</td>
</tr>
<tr>
<td>constant</td>
</tr>
<tr>
<td>ols.has.const, 59</td>
</tr>
<tr>
<td>correction</td>
</tr>
<tr>
<td>hcc, 46</td>
</tr>
<tr>
<td>covariance</td>
</tr>
<tr>
<td>makedata.corr, 54</td>
</tr>
</tbody>
</table>
INDEX

Sxy, 79
* covariation
  makedata.corr, 54
Sxy, 79
* datasets
  data.anscombe, 13
data.auto, 14
data.ballb, 15
data.burglary, 15
data.cars, 16
data.cobbdoug, 17
data.comp, 17
data.eu, 18
data.fertilizer, 20
data.filter, 21
data.govexpend, 22
data.icecream, 23
data.income, 23
data.insurance, 24
data.iv, 25
data.lifesat, 26
data.macro, 26
data.milk, 27
data.pharma, 28
data.printer, 29
data.regional, 30
data.rent, 31
data.savings, 31
data.sick, 32
data.software, 33
data.spurious, 34
data.tip, 35
data.tip.all, 36
data.trade, 36
data.unempl, 37
data.wage, 38
data.windscreen, 39
datasets, 39
* default value
  arguments, 6
* deformation
  def.exp, 41
def.log, 42
* deformed logarithm
  bc.model, 7
* delete
  new.session, 56
rm.all, 76
* desk
  datasets, 39
* endogeneity
  ivr, 48
* environment
  new.session, 56
rm.all, 76
* exponential
  def.exp, 41
* functional form
  reset.test, 75
* function
  arguments, 6
* heteroscedastic errors
  wh.test, 80
* heteroscedasticity
  lagk, 52
* heteroskedastic errors
  gq.test, 44
wh.test, 80
* heteroskedasticity
  bp.test, 9
gq.test, 44
hcc, 46
* htest
  bc.test, 7
* instrument variable
  ivr, 48
* intercept
  ols.has.const, 59
* lagged residuals
  lagk, 52
* lagged variables
  acc, 3
* lag
  lagk, 52
* level parameter
  ols.has.const, 59
* linear combination
  par.f.test, 63
* linear hypothesis
  par.f.test, 63
par.t.test, 64
* linear model
  ols, 57
ols.has.const, 59
ols.interval, 60
ols.predict, 61
par.f.test, 63
par.t.test, 64
* logarithm
  def.log, 42
* multicollinearity
  mc.table, 55
* non-constant error variance
  bp.test, 9
* non-stationarity
  roll.win, 77
* normal distribution
  jb.test, 51
* normality
  jb.test, 51
* null distribution
  ddw, 40
  pdw, 67
* objects
  new.session, 56
  rm.all, 76
* omitted variable bias
  repeat.sample, 72
* ordinary least squares
  ols, 57
  repeat.sample, 72
* plot
  makedata.bc, 53
  plot.desk, 68
* predicted values
  ols.predict, 61
* prediction band
  makedata.bc, 53
* prediction interval
  ols.interval, 60
  repeat.sample, 72
* print
  print.desk, 69
* prognostic Chow test
  pc.test, 66
* regression line
  makedata.bc, 53
* regression results
  plot.desk, 68
  print.desk, 69
* regression
  ols, 57
* remove
  new.session, 56
  rm.all, 76
* repeated samples
  repeat.sample, 72
* rolling window
  roll.win, 77
* scatter
  makedata.bc, 53
* simulation
  ar1sim, 4
  repeat.sample, 72
* specification
  reset.test, 75
* startup
  rprofile.add, 78
  rprofile.open, 78
* stationarity
  roll.win, 77
* structural break
  pc.test, 66
  qlr.cv, 70
  qlr.test, 71
* t-test
  par.t.test, 64
* test results
  plot.desk, 68
  print.desk, 69
* test
  jb.test, 51
* true regression parameters
  repeat.sample, 72
* unknown breakpoint
  qlr.cv, 70
  qlr.test, 71
* variance
  makedata.corr, 54
  Sxy, 79
* variation
  makedata.corr, 54
  Sxy, 79
* weak dependence
  roll.win, 77

acc, 3
acf, 4
ar1sim, 4
args, 6
arguments, 6

bc.model, 7
INDEX

bc.test, 7  
boxcox, 8  
bp.test, 9  
bptest, 11, 46, 80  
cochorc, 11  
data.anscombe, 13  
data.auto, 14  
data.ballb, 15  
data.burglary, 15  
data.cars, 16  
data.cobbdoug, 17  
data.comp, 17  
data.eu, 18  
data.fertilizer, 20  
data.filter, 21  
data.govexpend, 22  
data.icecream, 23  
data.income, 23  
data.insurance, 24  
data.iv, 25  
data.lifesat, 26  
data.macro, 26  
data.milk, 27  
data.pharma, 28  
data.printer, 29  
data.regional, 30  
data.rent, 31  
data.savings, 31  
data.sick, 32  
data.software, 33  
data.spurious, 34  
data.tip, 35, 36  
data.tip.all, 35, 36  
data.trade, 36  
data.unempl, 37  
data.wage, 38  
data.windscreen, 39  
datasets, 39  
ddw, 40, 43, 68  
def.exp, 41, 42  
def.log, 41, 42  
dw.test, 41, 42, 68  
gq.test, 44  
gqttest, 45  
hcc, 46  
hilu, 47  
ivr, 48  
jb.test, 51  
lagk, 4, 52  
makedata.bc, 53  
makedata.corr, 54  
mc.table, 55  
model.matrix.default, 48, 57  
new.session, 56  
ols, 57  
ols.has.const, 59  
ols.infocrit, 59  
ols.interval, 60  
ols.predict, 61  
par.f.test, 63  
par.t.test, 64  
pc.test, 66  
pdw, 41, 43, 67  
plot.desk, 68  
print.desk, 69  
qlr.cv, 70  
qlr.test, 71  
repeat.sample, 72  
reset.test, 75  
resettest, 76  
rm.all, 76  
roll.win, 77  
rprofile.add, 78  
rprofile.open, 78  
Sxy, 79  
wh.test, 11, 45, 46, 80