Package 's20x'

July 1, 2026

Version 3.3.0

Title Functions for University of Auckland Course STATS 201/208 Data Analysis

Description A set of functions used in teaching STATS 201/208 Data Analysis at the University of Auckland. The functions are designed to make parts of R more accessible to a large undergraduate population who are mostly not statistics majors.

Depends R (>= 4.0.0)

Suggests bootstrap, dafs, emmeans, formatR, knitr, markdown, testthat (>= 3.0.0)

Encoding UTF-8

Imports stats, graphics, grDevices, methods, GGally, ggplot2, nlme, rlang, rmarkdown, rstudioapi, tools, utils

License GPL-2 | file LICENSE

URL https://github.com/STATS-UOA/s20x

BugReports https://github.com/STATS-UOA/s20x/issues

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

NeedsCompilation no

Author Brant Deppa [aut] (Wrote the original R scripts this package is derived from),
James Curran [aut, cre] (Wrote the original R package. Current maintainer.),
Hannah Yun [ctb],
Rachel Fewster [ctb],
Russell Millar [ctb],
Ben Stevenson [ctb],
Andrew Balemi [ctb],
Chris Wild [ctb],
Sophie Jones [ctb],

Dineika Chandra [ctb],
Brendan McArdle [ctb]

Maintainer James Curran <j.curran@auckland.ac.nz>

Repository CRAN

Date/Publication 2026-07-01 06:50:02 UTC

Contents

airpass.df *International Airline Passengers*

Description

Number of international airline passengers (in thousands) recorded monthly from January 1949 to December 1960.

Format

A data frame with 144 rows and 4 variables:

passengers Monthly total number of international airline passengers (in thousands).

t Integer time index from 1 to 144.

month Month of observation as a factor with levels `Jan` to `Dec`.

year Year of observation as a factor with levels `1949` to `1960`.

anova.tslm *ANOVA tables for time series linear models*

Description

Produces analysis-of-variance-style tables for ‘tslm‘ objects.

Usage

```
## S3 method for class 'tslm'
anova(object, ..., verbose = FALSE)
```

Arguments

`object`	a fitted ‘tslm‘ object.
`...`	optional additional fitted model objects for model comparisons.
`verbose`	logical. For AR-error models, use ‘TRUE‘ to return the raw underlying [nlme::anova.gls()] output.

Details

For ordinary ‘tslm()‘ fits without autoregressive error terms, ‘anova()‘ returns the usual analysis of variance table from [stats::anova.lm()].

For AR-error models fitted through [nlme::gls()], the reported tests are Wald-style tests of model terms. These test whether each term contributes to the fitted mean model after allowing for the estimated autocorrelation structure. Because these models do not use the ordinary independent-error sum-of-squares decomposition, the compact table reports ‘Df‘, ‘F value‘, and ‘Pr(>F)‘, but does not report ‘Sum Sq‘ or ‘Mean Sq‘. Compare nested AR-error models with care: ‘verbose = TRUE‘ exposes the underlying ‘nlme‘ comparison output rather than recreating an ordinary ‘lm‘ ANOVA table.

Use ‘verbose = TRUE‘ to see the underlying [nlme::anova.gls()] output.

Value

An analysis-of-variance-style table.

Examples

```
data(beer.df)
fit = tslm(beer ~ t + ar(1), data = beer.df, time = t)
anova(fit)
```

`apples.df` *Apples Data*

Description

These data come from a classic long-term experiment conducted at the East Malling Research Station, Kent, which is the centre for research into apple growing in the U.K. Commercial apple trees consist of two parts grafted together. The lowest part, the *rootstock*, largely determines the size of the tree, while the upper part (the *scion*) determines the fruit characteristics. Rootstocks propagated by cuttings (i.e. asexually produced) were once thought to result in smaller trees than those propagated from seeds (i.e. sexually produced). This hypothesis was re-examined in an experiment begun in 1918. Several trees of each type of 16 types of rootstock were planted, all trees having the same scion. Rootstocks I-IX were asexually produced, while X-XVI were sexually produced. In the winter of 1933-4 a number of trees were removed to make room for more, and the data presented here consists of the above-ground weights of 104 trees felled in this period. No trees of types VIII, XI or XIV were felled. The description is adapted from Lee (1994). The data are from Andrews and Herzberg (1985).

Format

The data consist of a data frame with 104 observations on 4 variables.

Rootstock Factor giving the rootstock type (I, II, III, IV, IX, V, VI, VII, X, XII, XIII, XV, XVI).

Weight Integer Above-ground weight of tree (pounds, lb).

Weight_kg Numeric Above-ground weight of tree (kilograms, kg); `Weight_kg = Weight * 0.45359237`.

Propagated Factor giving the propagation method (cutting, seed).

References

Andrews, D. F. and Herzberg, A. M. (1985). *Data: A Collection of Problems from Many Fields for the Student and Research Worker*. New York: Springer.

Lee, A. J. (1994). *Data Analysis: An Introduction Based on R*. University of Auckland.

`arousal.df` *Changes in Pupil Size with Emotional Arousal*

Description

Data from an experiment to measure the effect of different images on emotional arousal, by measuring changes in pupil diameter. The experiment used 20 males and 20 females. Images included a nude man, nude woman, infant, and a landscape.

Format

A data frame with 160 observations on 3 variables.

arousal Numeric Change in the subject's pupil size.

gender Factor Subject's gender (female, male)

picture Factor Picture shown to subject (infant, landscape, nude female, nude male)

`autocorPlot` *Autocorrelation Plot*

Description

Plots current vs lagged residuals along with quadrants dividing these residuals about the value zero.

Usage

```
autocorPlot(fit, main = "Current vs Lagged residuals", ...)
```

Arguments

`fit`	output from the function 'lm()'.
`main`	the plot title.
`...`	extra parameters to be passed to the `plot` function.

Value

Plots current vs lagged residuals along with quadrants dividing these residuals about the value zero.

Note

`autocor.plot` is deprecated and no longer exported. Use `autocorPlot()` in new code.

Examples

```
data(airpass.df)
time = 1:144
airpass.fit = lm(passengers ~ time, data = airpass.df)
autocorPlot(airpass.fit)
```

---

beer.df *US Beer Production*

Description

Monthly United States beer production figures (in millions of 31-gallon barrels) for the period July 1970 to June 1978.

Format

A data frame with 96 rows and 4 variables:

beer Monthly beer production, expressed in megalitres (converted from millions of 31-US-gallon barrels; 1 million 31-gallon barrels is approximately equal to 117.35 megalitres).

t Integer time index from 1 to 96.

month Month of observation as a factor with levels `Jul`, `Aug`, `Sep`, `Oct`, `Nov`, `Dec`, `Jan`, `Feb`, `Mar`, `Apr`, `May`, `Jun`.

year Year of observation as a factor with levels `1970` to `1978`.

Note

The original primary source for this monthly beer-production series is not identified in the available package materials.

`body.df` *Body Image and Ethnicity Data*

Description

This dataset originates from a study conducted at the University of Auckland in the early 1990s by Dr. R.A. Marshall and colleagues from the Department of Psychology. The research explored how cultural background and ethnic identity influence body image perceptions within the specific context of Aotearoa New Zealand.

Format

A data frame with 246 observations on 8 variables.

ethnicity Factor Subject's ethnicity (Asian, Europn, Maori, Pacific)

married Factor Whether the subject is married (no, yes)

bodyim Factor Subject's rating of themself (slight.uw, right, slight.ow, mod.ow, very.ow)

sm.ever Factor Whether the subject has ever smoked (no, yes)

weight Numeric Weight in kg.

height Numeric Height in cm.

age Numeric Age in years.

stressgp Factor Stress level group (low, medium, high)

Details

The study specifically focused on a cohort of women who were generally "thin" (slightly underweight for their body size). This was designed to investigate whether body dissatisfaction and varying self-perceptions persisted even among individuals who already met or approached Western "thin" ideals, and how these perceptions differed across Asian, European, Māori, and Pacific ethnic groups.

Source

Marshall, R.A., Department of Psychology, University of Auckland.

References

Lee, A. J. (1994). Data Analysis: An Introduction Based on R. University of Auckland.

books.df *Books Data*

Description

This data consists of 50 sentence lengths from each of 8 books. The books "Disclosure" and "Rising Sun" were written by Michael Crichton, whilst the others "Four Past Midnight", "The Dark Half", " Eye of the Dragon", "The Shining", "The Stand" and "The Tommy-Knockers" were written by Stephen King. The pages and sentences were chosen using a multistage design where the pages were selected at random, and then sentences within each page were selected at random. These data were collected by James Curran.

Format

The data frame consists of 400 observations on 2 variables.

length Integer sentence length, measured as the number of words in the sentence.

book Factor giving the book from which the sentence was sampled (4.Past.Mid, Dark.Half, Disclosure, Eye.Drag, Rising.Sun, Shining, Stand, T.Knock).

boxqq *Deprecated box plots and normal quantile-quantile plots*

Description

'boxqq()' is deprecated and is no longer exported. It draws boxplots and normal quantile-quantile plots of 'x' for each level of the grouping variable 'g'.

Usage

```
boxqq(formula, ...)
```

Arguments

formula	A symbolic specification of the form x ~ g can be given, indicating the observations in the vector x are to be grouped according to the levels of the factor g. NA's are allowed in the data.
...	Arguments to be passed to methods, such as graphical parameters (see par).

Value

Returns the plot.

Note

This is a legacy teaching helper retained for compatibility with older course material. New teaching material should prefer current diagnostic plotting workflows.

bursary.df *Bursary Results for Auckland Secondary Schools*

Description

Data for the 2001 Bursary results for 75 secondary schools in the Auckland area. For each school the decile rating of the school is recorded along with the percentage of eligible students who gain a B Bursary or better.

Format

A data frame with 75 observations on 2 variables.

decile Numeric Decile rating of the school.

pass.rate Numeric percentage of eligible students who gained a B Bursary or better.

butterfat.df *Butterfat Data*

Description

This data gives the mean percentage of butterfat produced by different Canadian pure-bred diary cattle. There are five different breeds and two age groups, two years old and greater than five years old. For each combination of breed and age, there are measurements for 10 cows.

Format

A data frame with 100 observations on 3 variables.

Butterfat Numeric mean percentage of butterfat per cow.

Breed Factor giving the cattle breed (ayrshire, canadian, guernesy, holst.fres, jersey).

Age Factor giving the age group (2yo, mature).

Source

A Handbook of Small Data Sets

References

Hand, D.J., Daly, F., Lunn, A.D., McConway, K.J. and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. Boca Raton, Florida: Chapman and Hall/CRC.

Sokal, R.R. and Rohlf, F.J. (1981). *Biometry*, 2nd edition. San Francisco: W.H. Freeman, 368.

camplake.df *Age and Length of Camp Lake Bluegills*

Description

66 bluegills were captured from Camp Lake, Minnesota. For each bluegill we have the length of the fish, its age in years and its age in scale radius.

Format

A data frame with 66 observations on 3 variables.

Age Numeric age of the fish, in years.

Scale.Radius Numeric radius of the key scale, in hundredths of a millimetre.

Length Numeric length at capture, in millimetres.

casestudy *Render a case study to HTML*

Description

Renders a specified case study R Markdown file shipped with the package to HTML and optionally opens it in a web browser.

Usage

```
casestudy(
  id,
  output_dir = tempfile("s20x_case_study_"),
  open = interactive(),
  quiet = TRUE,
  ...
)

cs(...)
```

Arguments

id	A case study identifier. Flexible formats are accepted, including "CS9_2", "CS9.2", "9_2", or "9.2".
output_dir	Directory where the rendered HTML file should be written. Defaults to a temporary directory. This legacy argument is retained for compatibility; new code may use the camelCase outputDir alias through ....
open	Logical; if TRUE (default), open the rendered HTML file in the default web browser.

quiet	Logical; passed to rmarkdown::render() to suppress output.
...	Additional arguments passed to rmarkdown::render(). Also supports outputDir, a camelCase alias for output_dir.

Details

Case studies are expected to live in inst/case_studies and to be named using the pattern CS<chapter>_<number>.Rmd (for example, CS9_2.Rmd).

The case study is rendered on demand using rmarkdown::render(). Figures and other outputs are generated at render time; users therefore need any required packages installed for the selected case study.

The rendered HTML file is returned invisibly.

Value

Invisibly returns the path to the rendered HTML file.

Examples

```
if (interactive()) {
  casestudy("CS9_2")
  casestudy("9.2")
  casestudy("9_2", outputDir = tempdir())
  cs("9_2")
}
```

---

chalk.df *Chalk Data*

Description

These data involve 11 laboratories and 2 brands of chalk. The laboratories tested the density of the chalk. The main interest was whether the different laboratories yielded the same density for the two different types of chalk.

Format

A data frame with 66 observations on 3 variables.

Density Numeric density of the chalk.

Lab Integer laboratory identifier.

Chalk Factor giving the chalk brand tested (A, B).

ciReg *Confidence Intervals for Regression models*

Description

Calculates and prints the confidence intervals for the fitted model.

Usage

```
ciReg(fit, conf.level = 0.95, print.out = TRUE)
```

Arguments

`fit`	an object of class `lm`, i.e. the output from `lm`.
`conf.level`	confidence level of the intervals.
`print.out`	if `TRUE`, print out the output on the screen.

Value

The function returns a two-column matrix containing the upper and lower endpoints of the intervals.

See Also

`lm`, `summary`, `anova`.

Examples

```
##Peruvian Indians data
data(peru.df)
fit = lm(BP ~ age + years + weight + height, data = peru.df)
ciReg(fit)
```

computer.df *Computer Questionnaire*

Description

Data from a test to see if a questionnaire was properly designed. The questionnaire measures managers' technical knowledge of computers. The test has 19 managers complete the questionnaire as well as rate their own technical expertise.

Format

A data frame with 19 observations on 2 variables.

score Numeric questionnaire score.

selfassess Ordered factor giving the self-assessed level of expertise (1 = low, 2 = medium, 3 = high).

`cooks20x` *Cook's distance plot*

Description

Draws a Cook's distance plot.

Usage

```
cooks20x(
  x,
  main = "Cook's Distance plot",
  xlab = "observation number",
  ylab = "Cook's distance",
  line = c(0.5, 1.2, 2),
  cex.labels = 1,
  axisOpts = list(xAxis = TRUE, yAxisTight = FALSE),
  ...
)
```

Arguments

`x`	an object of class `lm`, usually obtained by using the `lm` function.
`main`	the plot title
`xlab`	the x-axis title.
`ylab`	the y-axis title.
`line`	a vector of length 3 controlling the distances of the plot title, the x-axis title and the y-axis title from the axis in line units.
`cex.labels`	a factor controlling the font size of the labels on suspected high influence points.
`axisOpts`	a list of additional arguments that can be used to control the axes. At this point this list only contains one element `xAxis` which is logical. If `xAxis == TRUE` then the x-axis will be displayed, and clearly, if it is `FALSE`, then it will not.
`...`	additional arguments are passed to `plot` and may provide some extra flexibility.

Value

Returns the plot and identifies the three highest Cook's values

Examples

```
# Peruvian Indians data
data(peru.df)
peru.fit = lm(BP ~ age + years + I(years^2) + weight + height, data = peru.df)
cooks20x(peru.fit)
```

`course.df` *Stats 20x Summer School Data*

Description

Data from a summer school Stats 20x course. Each observation represents a single student.

Format

A data frame with 146 observations on 15 variables.

Grade Factor Final grade for the course (A, B, C, D)

Pass Factor Passed the course (No, Yes)

Exam Numeric Mark in the final exam.

Degree Factor Degree enrolled in (BA, BCom, BSc, Other)

Gender Factor Gender (Female, Male)

Attend Factor Regularly attended class (No, Yes)

Assign Numeric Assignment mark.

Test Numeric Test mark.

B Numeric Mark for the short answer section of the exam.

C Numeric Mark for the long answer section of the exam.

MC Numeric Mark for the multiple choice section of the exam.

Colour Factor Colour of the exam booklet (Blue, Green, Pink, Yellow)

Stage1 Factor Stage one grade (A, B, C)

Years.Since Numeric Number of years since doing Stage 1.

Repeat Factor Repeating the paper (No, Yes)

course2way.df — *Exam Mark, Gender and Attendance for Stats 20x Summer School Students*

Description

Data from a summer school Stats 20x course. Each observation represents a single student. It is of interest to see if there is a relationship between a student's final examination mark and both their gender and whether they regularly attend lectures.

Format

A data frame with 40 observations on 3 variables.

Exam Numeric Final exam mark (out of 100)

Gender Factor Gender (Female, Male)

Attend Factor Regularly attended or not (No, Yes)

crossFactors — *Crossed Factors*

Description

Computes a factor that has a level for each combination of the factors 'fac1' and 'fac2'.

Usage

```
crossFactors(x, fac2 = NULL, ...)

## Default S3 method:
crossFactors(x, fac2 = NULL, ...)

## S3 method for class 'formula'
crossFactors(formula, fac2 = NULL, data = NULL, ...)
```

Arguments

x	the name of the first factor or a formula in the form ~ fac1 * fac2
fac2	the name of the second factor - ignored if x is a formula.
...	Optional arguments
formula	a formula in the form ~ fac1 * fac2
data	an optional data frame in which to evaluate the formula

Value

Returns a vector containing the factor which represents the interaction of the given factors.

Methods (by class)

- `crossFactors(default)`: Crossed Factors
- `crossFactors(formula)`: Crossed Factors

Note

This function actually returns a `factor` now instead of a character string, so coercion into a `factor` is no longer necessary.

See Also

`factor`.

Examples

```
## arousal data:
data(arousal.df)
gender.picture = crossFactors(arousal.df$gender, arousal.df$picture)
gender.picture

## arousal data:
data(arousal.df)
gender.picture = crossFactors(~ gender * picture, data = arousal.df)
gender.picture
```

`crosstabs` *Crosstabulation of two variables*

Description

Produces a 2-way table of counts and the corresponding chi-square test of independence or homogeneity.

Usage

```
crosstabs(formula, data)
```

Arguments

`formula`	a symbolic description of the model to be fit: ~ fac1 + fac2; where fac1 and fac2 are vectors to be crosstabulated and treated internally as factors.
`data`	an optional data frame containing the variables in the model.

Value

Invisibly returns an object of class `ct.20x`, which is a list containing the following components:

`row.props`	a matrix of row proportions, i.e. cell counts divided by row marginals.
`col.props`	a matrix of column proportions, i.e. cell counts divided by column marginals.
`whole.props`	a matrix of whole-table proportions.
`Totals`	a matrix containing the cell counts and the marginal totals.
`exp`	a matrix of expected counts from the chi-square calculation.
`chi`	a matrix of cell contributions to the chi-square statistic.

Note

This is a legacy teaching helper retained for compatibility with older course material. New code should usually prefer `table()` and `chisq.test()` directly, or a purpose-built teaching wrapper.

Examples

```
##body image data:
data(body.df)
crosstabs(~ ethnicity + married, body.df)
```

---

diamonds.df *Prices and Weights of Diamonds*

Description

Prices of ladies' diamond rings from a Singaporean retailer and the weight of their diamond stones.

Format

A data frame with 48 observations on 2 variables.

price Numeric Price of ring (Singapore dollars)

weight Numeric Weight of Diamond (carats)

displayPairs *Display within-level pairwise comparisons for saturated two-way ANOVA model.*

Description

Displays within-level pairwise comparisons from a two-way ANOVA with interactions. Note that this is just a display function: it ignores any cross-level pairs included in `allpairs`, even though these will have contributed to the computations for the Tukey adjustments. The purpose is just to organise the output from `emmeans` into a more convenient format.

Usage

```
displayPairs(allpairs, levels1, levels2, brief = TRUE, asDF = FALSE)
```

Arguments

`allpairs`	pairwise output from a command like `pairs`. See details for a longer explanation.
`levels1`	a character string specifying which within-level comparisons from `factor1` are wanted, and in which order.
`levels2`	a character string specifying which within-level comparisons from `factor2` are wanted, and in which order.
`brief`	either `TRUE` or `FALSE`. If `TRUE` then the information displayed will be more succinct.
`asDF`	either `TRUE` or `FALSE` specifying whether to return a `data.frame` of results or just to display the output.

Details

`allpairs` is a pairwise output from a command like `pairs(emmeans(fit, ~factor1 * factor2))`. If `allpairs` is not already a `data.frame` it will be converted to a `data.frame` within this function. It must contain a column called `contrast` with text descriptions like `'lev1 lev2 - lev3 lev4'` etc. `levels1` and `levels2` are character strings specifying which within-level comparisons are wanted, and in which order. They must match the order specified in `emmeans`, so if using `emmeans(fit, ~factor1 * factor2)` then `levels1` must belong to `factor1` and `levels2` must belong to `factor2`. All this function does is to pick out the rows of `allpairs` with the requested contrasts, so if there are no contrasts of the requested format (e.g. because `levels1` and `levels2` have been switched) it will output a blank list. If `brief = TRUE`, columns labelled `df`, `SE`, and `t.ratio` or `z.ratio` will be removed for a more succinct display. If `asDF = TRUE`, the output is returned as a data-frame suitable for further manipulation, whereas if `asDF = FALSE` it is returned as a list for display only.

Author(s)

Rachel Fewster

Examples

```
## Fit a two-way ANOVA to the arousal data in arousal.df.
## The factors are gender (female, male) and picture shown to
## subject (infant, landscape, nude.f, nude.m):
data(arousal.df)
arousal.fit = lm(arousal ~ gender *  picture, data = arousal.df)

## Create all pairwise comparisons using emmeans, if available.
if (requireNamespace("emmeans", quietly = TRUE)) {
    emmeansFun = getExportedValue("emmeans", "emmeans")
    arousal.allpairs = pairs(
        emmeansFun(arousal.fit, ~ gender * picture),
        infer = TRUE
    )

    ## Display only the within-level comparisons:
    displayPairs(
        arousal.allpairs,
        levels1 = c("female", "male"),
        levels2 = c("infant", "landscape", "nude.f", "nude.m")
    )
}
```

---

eovcheck *Testing for equality of variance plot*

Description

Plots the residuals versus the fitted (or predicted) values from a linear model. A horizontal line is drawn at y = 0, reflecting the fact that we expect the residuals to have a mean of zero. An optional lowess line is drawn if smoother is set to TRUE. This can be useful in determining whether a trend still exists in the residuals. An optional pair of lines is drawn at +/- 2 times the standard deviation of the residuals - which is estimated from the Residual Mean Sqare (Within group mean square = WGMS). This can be useful in highlighting potential outliers. If the model has one or two factors and no continous variables, i.e. if it is a oneway or twoway ANOVA model, and `levene = TRUE` then the P-value from Levene's test for equality variance is displayed in the top left hand corner, as long as the number of observations per group exceeds two.

Usage

```
eovcheck(x, ...)

## S3 method for class 'formula'
eovcheck(
  x,
  data = NULL,
  xlab = "Fitted values",
```

```
  ylab = "Residuals",
  col = NULL,
  smoother = FALSE,
  twosd = FALSE,
  levene = FALSE,
  engine = c("base", "ggplot2"),
  ...
)

## S3 method for class 'lm'
eovcheck(
  x,
  smoother = FALSE,
  twosd = FALSE,
  levene = FALSE,
  engine = c("base", "ggplot2"),
  ...
)
```

Arguments

x	A linear model formula. Alternatively, a fitted lm object from a linear model.
...	Optional arguments passed to the base plotting engine. Extra arguments are currently ignored by the ggplot2 engine.
data	A data frame in which to evaluate the formula.
xlab	a title for the x axis: see title.
ylab	a title for the y axis: see title.
col	a colour for the lowess smoother line.
smoother	if TRUE then a smoothed lowess line will be added to the plot
twosd	if TRUE then horizontal dotted lines will be drawn at +/-2sd
levene	if TRUE then the P-value from Levene's test for equality of variance is displayed
engine	plotting engine to use. The default, "base", preserves the original base graphics output. Use "ggplot2" for an optional ggplot2 object.

Details

The default base graphics engine preserves the original teaching plot and draws directly on the active graphics device. The optional ggplot2 engine is intended for users who want a reusable plot object for reports or further customisation; it requires **ggplot2** to be installed and returns a ggplot object instead of drawing a base graphics side effect.

Value

Draws the residual-versus-fitted diagnostic plot when using the base engine. With engine = "ggplot2", returns a ggplot object.

See Also

levene.test

Examples

```
# one way ANOVA - oysters
data(oysters.df)
oyster.fit = lm(Oysters ~ Site, data = oysters.df)
eovcheck(oyster.fit)

# Same model as the previous example, but using eovcheck.formula
data(oysters.df)
eovcheck(Oysters ~ Site, data = oysters.df)


# A two-way model without interaction
data(soyabean.df)
soya.fit = lm(yield ~ planttime + cultivar, data = soyabean.df)
eovcheck(soya.fit)

# A two-way model with interaction
data(arousal.df)
arousal.fit = lm(arousal ~ gender * picture, data = arousal.df)
eovcheck(arousal.fit)

# A regression model
data(peru.df)
peru.fit = lm(BP ~ height + weight + age + years, data = peru.df)
eovcheck(peru.fit)


# A time series model
data(airpass.df)
t = 1:144
month = factor(rep(1:12, 12))
airpass.df = data.frame(passengers = airpass.df$passengers, t = t, month = month)
airpass.fit = lm(log(passengers)[-1] ~ t[-1] + month[-1]
                 + log(passengers)[-144], data  = airpass.df)
eovcheck(airpass.fit)

# Optional ggplot2 engine for reusable plot objects
if (requireNamespace("ggplot2", quietly = TRUE)) {
  eovPlot = eovcheck(oyster.fit, engine = "ggplot2")
  class(eovPlot)

  eovcheck(peru.fit, engine = "ggplot2", smoother = TRUE)
  eovcheck(oyster.fit, engine = "ggplot2", twosd = TRUE, levene = TRUE)
}
```

`estimateContrasts` *Contrast Estimates*

Description

Calculates and prints Tukey multiple confidence intervals for contrasts in one or two-way ANOVA.

Usage

```
estimateContrasts(
  contrast.matrix,
  fit,
  row = TRUE,
  alpha = 0.05,
  L = NULL,
  FUN = identity
)
```

Arguments

`contrast.matrix`	A matrix of contrast coefficients. Separate rows of the matrix contain the contrast coefficients for that particular contrast, and a column for each level of the factor.
`fit`	Output from the [lm()] function.
`row`	If 'TRUE', and the ANOVA is two-way, then contrasts in the row effects are printed, otherwise contrasts in the column effects are printed. Ignored if the ANOVA is one-way.
`alpha`	The nominal error rate for the multiple confidence intervals.
`L`	Number of contrasts. If 'NULL', 'L' will be set to the number of rows in the contrast matrix, otherwise 'L' will be as specified.
`FUN`	Optional function to be applied to estimates and confidence intervals. Typically used for back-transformation operations.

Value

Returns a matrix whose rows correspond to the different contrasts being estimated and whose columns correspond to the point estimate of the contrast, the Tukey lower and upper limits of the confidence interval, the unadjusted p-value, and the Tukey and Bonferroni p-values.

See Also

[summary1way()], [summary2way()], [multipleComp()]

Examples

```
## computer data:
data(computer.df)
computer.df = within(computer.df, {selfassess = factor(selfassess)})
computer.fit = lm(score ~ selfassess, data = computer.df)
contrast.matrix = matrix(c(-1 / 2, -1 / 2, 1), byrow = TRUE, nrow = 1, ncol = 3)
contrast.matrix
estimateContrasts(contrast.matrix, computer.fit)
```

---

fire.df *Fire Damage and Distance from the Fire Station*

Description

House damage and distance from the fire station, of 15 house fires. Data collected by an insurance company for homes in a particular area.

Format

A data frame with 15 observations on 3 variables.

damage Numeric Damage (1000s of dollars)

distance Numeric Distance from the fire station (miles)

distance_km Numeric Distance from the fire station (kilometres); `distance_km = distance * 1.60934`.

---

freq1way *Analysis of 1-dimensional frequency tables*

Description

If hypothprob is absent: prints confidence intervals for the true proportions, a Chi-square test for uniformity, confidence intervals for differences in proportions (with no corrections for multiple comparisons), and plots the proportions.

Usage

```
freq1way(
  counts,
  hypothprob,
  conf.level = 0.95,
  addCIs = TRUE,
  digits = 4,
  arrowwid = 0.1,
  estimated = 0
)
```

Arguments

counts	A 1-way frequency table as produced by table.
hypothprob	If present, a set of probabilities to test the cell counts against.
conf.level	confidence level for the confidence interval, expressed as a decimal.
addCIs	If true, adds confidence limits to plot of sample proportions.
digits	used to control rounding of printout.
arrowwid	controls width of arrowheads.
estimated	default is 0. Subtracted from the df for the Chi-square test.

Details

If hypothprob is present: prints confidence intervals for the true proportions, a Chi-square test for the hypothesised probabilities, and plots the sample proportions (with attached confidence limits) alongside the corresponding hypothesised probabilities.

Value

An invisible list containing the following components:

CIs	a matrix containing the confidence intervals.
exp	a vector of the expected counts.
chi	a vector of the components of Chi-square.

Note

These confidence intervals have been Bonferroni adjusted for multiple comparisons. This is a legacy teaching helper retained for compatibility with older course material.

Examples

```
##Body image data:
data(body.df)
eth.table = with(body.df, table(ethnicity))
freq1way(eth.table)
freq1way(eth.table,hypothprob=c(0.2,0.4,0.3,0.1))
```

fruitfly.df *Fruitfly Data*

Description

This data gives fecundity for female fruitflies, Drosophila melanogaster. The fecundity is the number of eggs laid, per day, for the fruitfly's first 14 days of life. There are three strains: A control group, NS, Nonselected Strain, as well as RS, a strain bred for resistance to DDT and SS, a strain bred for susceptibility to DDT. Each strain contains 25 measurements. It is of interest to compare the level of fecundity across strains.

Format

A data frame with 75 observations on 2 variables.

fecundity Numeric Number of eggs laid, per day, per fruitfly.

strain Factor Strain of fruitfly (NS, RS, SS)

Source

A Handbook of Small Data Sets

References

Hand, D.J., Daly, F., Lunn, A.D., McConway, K.J. and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. Boca Raton, Florida: Chapman and Hall/CRC.

Sokal, R.R. and Rohlf, F.J. (1981). *Biometry*, 2nd edition. San Francisco: W.H. Freeman, 239.

getVersion *s20x package version number*

Description

Returns the version number of the s20x package. This is useful if a student has problems running commands and the maintainer needs to check the version number.

Usage

```
getVersion()
```

Examples

```
getVersion()
```

house.df *Sale and Advertised Prices of Houses*

Description

A random sample of 100 houses recently sold in Mt Eden, Auckland. For each house we have the advertised price and the actual sale price.

Format

A data frame with 100 observations on 2 variables.

advertised.price Numeric Advertised price (dollars)

sell.price Numeric Final sale price (dollars)

incomes.df *Mean Family Incomes*

Description

Random sample of 152 families giving their mean income (1000s of dollars). The sample was taken by an advertising agency over their area of operations.

Format

A data frame with 152 observations on 1 variable.

incomes Numeric mean family income, in thousands of dollars.

interactionPlots *Interactions Plot for Two-way Analysis of Variance*

Description

Displays data with intervals for each combination of the two factors and shows the mean differences between levels of the first factor for each level of the second factor. Note that there should be more than one observation for each combination of factors.

Usage

```
interactionPlots(y, ...)

## Default S3 method:
interactionPlots(
  y,
  fac1 = NULL,
  fac2 = NULL,
  xlab = NULL,
  xlab2 = NULL,
  ylab = NULL,
  data.order = TRUE,
  exlim = 0.1,
  jitter = 0.02,
  conf.level = 0.95,
  interval.type = c("tukey", "hsd", "lsd", "ci"),
  pooled = TRUE,
  tick.length = 0.1,
  interval.distance = 0.2,
  col.width = 2/3,
  xlab.distance = 0.1,
  xlen = 1.5,
  ylen = 1,
  ...
)

## S3 method for class 'formula'
interactionPlots(
  y,
  data = NULL,
  xlab = NULL,
  xlab2 = NULL,
  ylab = NULL,
  data.order = TRUE,
  exlim = 0.1,
  jitter = 0.02,
  conf.level = 0.95,
  interval.type = c("tukey", "hsd", "lsd", "ci"),
  pooled = TRUE,
  tick.length = 0.1,
  interval.distance = 0.2,
  col.width = 2/3,
  xlab.distance = 0.1,
  xlen = 1.5,
  ylen = 1,
  ...
)
```

Arguments

y	either a formula of the form: y~fac1+fac2 where y is the response and fac1 and fac2 are the two explanatory variables used as factors, or a single response vector
...	optional arguments.
fac1	if 'y' is a vector, then fac1 contains the levels of factor 1 which correspond to the y value
fac2	if 'y' is a vector, then fac2 contains the levels of factor 2 which correspond to the y value
xlab	an optional label for the x-axis. If not specified the name of fac1 will be used.
xlab2	an optional label for the lines. If not specified the name of fac2 will be used.
ylab	An optional label for the y-axis. If not specified the name of y will be used.
data.order	if TRUE the levels of fac1 and fac2 will be set to unique(fac1) and unique(fac2) respectively.
exlim	provide extra limits.
jitter	the amount of horizontal jitter to show in the plot. The actual jitter is determined as the function is called, and will likely be different each time the function is used.
conf.level	confidence level of the intervals.
interval.type	four options for intervals appearing on plot: 'tukey', 'hsd', 'lsd' or 'ci'.
pooled	two options: pooled or unpooled standard deviation used for plotted intervals.
tick.length	size of tick, in inches.
interval.distance	distance, as a fraction of the column width, between the points and interval. This is in addition to the extra space allocated for the jitter.
col.width	width of a factor 'column', as a fraction of the space between the centres of two columns.
xlab.distance	distance of x-axis labels from bottom of plot, as a fraction of the overall height of the plot.
xlen, ylen	character interspacing factor for horizontal (x) and vertical (y) spacing of the legend.
data	an optional data frame containing the variables in the model.

Methods (by class)

- `interactionPlots(default)`: Interactions Plot for Two-way Analysis of Variance
- `interactionPlots(formula)`: Interactions Plot for Two-way Analysis of Variance

See Also

summary2way.

Examples

```
data(arousal.df)
interactionPlots(arousal ~ gender + picture, data = arousal.df)

## This usage is deprecated.
with(arousal.df, interactionPlots(arousal, gender, picture))
```

---

lakemary.df *Ages and Lengths of Lake Mary Bluegills*

Description

The ages and lengths of 78 bluegills captured from Lake Mary, Minnesota.

Format

A data frame with 78 observations on 2 variables.

Age Numeric Age of the fish (years)

Length Numeric Length at capture (mm)

---

larain.df *Los Angeles Rainfall*

Description

Annual rainfall (in inches) for Los Angeles from 1908 to 1973.

Format

A data frame with 66 rows and 4 variables:

LA.Rain Annual rainfall in Los Angeles, measured in inches.

rain_mm Annual rainfall in Los Angeles, measured in millimetres (mm); `rain_mm = LA.Rain * 25.4`.

t Integer time index from 1 to 66.

year Year of observation as an integer from 1908 to 1973.

layout20x *Layout*

Description

Allows a ‘numRows‘ by ‘numCols‘ matrix of plots to be displayed in a single plot. If the function is called with no arguments, then the plotting device layout will be reset to a single plot.

Usage

```
layout20x(numRows = 1, numCols = 1)
```

Arguments

numRows	Number of rows in the plot array.
numCols	Number of columns in the plot array.

Value

No return value.

Note

This is a legacy convenience wrapper retained for compatibility with older teaching material. New code can use `par(mfrow = ...)` directly.

Examples

```
data(course.df)
layout20x(1, 2)
stripchart(course.df$Exam)
boxplot(course.df$Exam)
```

levene.test *Levene test for the ANOVA Assumption*

Description

Perform a Levene test for equal group variances in both one-way and two-way ANOVA. A table with the results is (normally) displayed.

Usage

```
levene.test(formula, data, digit = 5, show.table = TRUE)
```

Arguments

formula	a symbolic description of the model to be fitted: response ~ fac1 + fac2.
data	an optional data frame containing the variables in the model.
digit	the number of decimal places to display.
show.table	If this argument is FALSE then the output will be suppressed

Value

A list with the following elements:

df	degrees of freedom.
ss	sum squares.
ms	mean squares.
f.value	F-statistic value.
p.value	P-value.

See Also

crossFactors, anova.

Examples

```
##
data(computer.df)
levene.test(score ~ factor(selfassess), computer.df)
```

---

listCaseStudies *List available case studies*

Description

Lists all case study R Markdown files shipped with the package and prints them as a formatted text table.

Usage

```
listCaseStudies()

listCS()

lcs()
```

Details

Case studies are expected to live in `inst/case_studies` and to be named using the pattern `CS<chapter>_<number>.Rmd` (e.g. `CS9_2.Rmd`).

The table has two columns: `File` (the case study identifier) and `Title` (extracted from the YAML header). Case studies are listed in numerical order, not alphabetical order.

The function invisibly returns a character vector of case study identifiers.

Value

Invisibly returns a character vector of case study identifiers.

Examples

```
if (interactive()) {
  listCaseStudies()
  ids = listCaseStudies()
}
```

---

mazda.df *Year and Price of Mazda Cars*

Description

Prices and ages of 124 Mazda cars collected from the Melbourne Age newspaper in 1991.

Format

A data frame with 124 observations on 2 variables.

price Numeric Price (Australian dollars)

year Numeric Year of manufacture.

---

mening.df *Monthly Notifications of Meningococcal Disease*

Description

This data shows the monthly number of notifications meningococcal disease in New Zealand from January 1990 to December 2001.

Format

A data frame with 144 observations on 3 variables.

Month Factor giving the month of notification.

Year Factor giving the year of notification.

mening Numeric number of notifications of meningococcal disease.

mergers.df *Merger Days*

Description

A random selection of 38 consummated mergers from the USA, 1982, giving the number of days between the date the merger was announced and the date the merger became effective.

Format

A data frame with 38 observations on 1 variable.

mergerdays Numeric number of days between the merger announcement and the effective date.

modcheck *Deprecated model checking plots*

Description

'modcheck()' is deprecated and is no longer exported. It plots four model checking plots: residuals versus fitted values, a normal Q-Q plot, a histogram of residuals with a normal distribution superimposed, and a Cook's distance plot.

Usage

```
modcheck(x, ...)
```

Arguments

x	a vector of observations, or the residuals from fitting a linear model. Alternatively, a fitted `lm` object. If `x` is a single vector, then the implicit assumption is that the mean (or null) model is being fitted, i.e. `lm(x ~ 1)` and that the data are best summarised by the sample mean.
...	additional parameters. Included for future flexibility, but unsure how this might be used currently.

Value

Draws the selected model checking plots for teaching diagnostics. The function is called for its plotting side effects and does not provide a stable data return object.

`modelcheck` *Model checking plots*

Description

Draw the teaching diagnostic plots used by older ‘s20x‘ workflows. ‘modelcheck()‘ is retained as an exported compatibility helper for model checking, while newer teaching material may use focused diagnostic helpers such as [eovcheck()], [normcheck()], and [cooks20x()] directly.

Usage

```
modelcheck(x, ...)

## S3 method for class 'lm'
modelcheck(
  x,
  which = 1:3,
  mar = c(3, 4, 1.5, 4),
  engine = c("base", "ggplot2"),
  ...
)
```

Arguments

`x`	The fitted model.
`which`	The plot(s) to be drawn. Residuals versus fitted values (`which = 1`), histogram and Q-Q plot of residuals (`which = 2`), and Cook's distance plot (`which = 3`).
`mar`	Margins applied to each selected plot. Ignored by the ggplot2 engine.
`engine`	plotting engine to use. The default, `"base"`, preserves the original base graphics output. Use `"ggplot2"` for optional ggplot2 objects.
`...`	any other arguments to pass to `plot` for the base engine. Extra arguments are currently ignored by the ggplot2 engine.

Details

The default base graphics engine preserves the original teaching plots and draws directly on the active graphics device. The optional ggplot2 engine is intended for users who want reusable plot objects for reports or further customisation; it requires **ggplot2** to be installed and returns ggplot objects instead of drawing base graphics side effects.

Value

Draws diagnostic plots for teaching model checking when using the base engine. With `engine = "ggplot2"`, returns a ggplot object for a single selected plot, or a named list of ggplot objects for multiple selected plots.

Examples

```
data(peru.df)
lmFit = lm(BP ~ weight, data = peru.df)

# Plot residuals versus fitted values only
modelcheck(lmFit, 1)

# Plot residuals versus fitted values, histogram, and Q-Q plot
modelcheck(lmFit, 1:2)

# Plot all diagnostics
modelcheck(lmFit)

# Optional ggplot2 engine for reusable plot objects
if (requireNamespace("ggplot2", quietly = TRUE)) {
  diagnosticPlots = modelcheck(lmFit, engine = "ggplot2")
  names(diagnosticPlots)

  modelcheck(lmFit, which = 1, engine = "ggplot2")
  modelcheck(lmFit, which = 2, engine = "ggplot2")
  modelcheck(lmFit, which = 3, engine = "ggplot2")
}
```

---

mozart.df *Length of Mozart's Movements*

Description

Length of movements from 11 of Mozart's early symphonies and 11 of his late symphonies.

Format

A data frame with 88 observations on 3 variables.

Time Numeric Time of each movement (seconds)

Movement Factor Movement (M1, M2, M3, M4)

Period Factor Period that the symphony was written (early, late)

---

multipleComp *Multiple Comparisons*

Description

Calculates and prints the estimate, multiple 95% confidence intervals, unadjusted, Tukey and Bonferroni p-values for all possible differences in means in a one-way ANOVA.

Usage

```
multipleComp(fit, conf.level = 0.95, FUN = identity)
```

Arguments

fit	Output from the command [lm()].
conf.level	Confidence level for the confidence interval, expressed as a percentage.
FUN	Optional function to be applied to estimates and confidence intervals. Typically used for back-transformation operations.

Value

Returns a list of estimates, confidence intervals and p-values.

Examples

```
## computer data
data(computer.df)
fit = lm(score ~ factor(selfassess), data = computer.df)
multipleComp(fit)

## butterfat data
data("butterfat.df")
fit = lm(log(Butterfat) ~ Breed, data = butterfat.df)
multipleComp(fit, FUN = exp)
```

---

nail.df *Nail Polish Data*

Description

These data were collected to determine whether quick drying nail polish or regular nail polish dried faster. The time for each type of nail polish to dry was recorded.

Format

A data frame with 60 observations on 2 variables.

polish Factor Type of polish (Regular, Quick)

dry Integer Time (in seconds) for the polish to dry.

normcheck *Testing for normality plot*

Description

Plots two plots side by side. First, it draws a normal Q-Q plot of the residuals, along with a line with intercept equal to the mean of the residuals and slope equal to the standard deviation of the residuals. If `shapiro.wilk = TRUE`, the P-value from the Shapiro-Wilk test for normality is shown in the top-left corner of the Q-Q plot. Second, it draws a histogram of the residuals. A normal distribution is fitted and superimposed over the histogram. Note: if you want to leave the x-axis blank in the histogram then use `xlab = c("Theoretical Quantiles", " ")` , i.e. leave a space between the quotes. If you do not leave a space, information will be extracted from `x`.

Usage

```
normcheck(x, ...)

## Default S3 method:
normcheck(
  x,
  xlab = c("Theoretical Quantiles", ""),
  ylab = c("Sample Quantiles", ""),
  main = c("", ""),
  col = "light blue",
  bootstrap = FALSE,
  B = 5,
  bpch = 3,
  bcol = "lightgrey",
  shapiro.wilk = FALSE,
  whichPlot = 1:2,
  usePar = TRUE,
  engine = c("base", "ggplot2"),
  ...
)

## S3 method for class 'lm'
normcheck(
  x,
  xlab = c("Theoretical Quantiles", ""),
  ylab = c("Sample Quantiles", ""),
  main = c("", ""),
  col = "light blue",
  bootstrap = FALSE,
  B = 5,
  bpch = 3,
  bcol = "lightgrey",
  shapiro.wilk = FALSE,
```

```
  whichPlot = 1:2,
  usePar = TRUE,
  engine = c("base", "ggplot2"),
  ...
)

## S3 method for class 'tslm'
normcheck(
  x,
  xlab = c("Theoretical Quantiles", ""),
  ylab = c("Sample Quantiles", ""),
  main = c("", ""),
  col = "light blue",
  bootstrap = FALSE,
  B = 5,
  bpch = 3,
  bcol = "lightgrey",
  shapiro.wilk = FALSE,
  whichPlot = 1:2,
  usePar = TRUE,
  residualType = "normalised",
  engine = c("base", "ggplot2"),
  ...
)
```

Arguments

x	the residuals from fitting a linear model. Alternatively, a fitted lm object.
...	additional arguments which are passed to both qqnorm and hist for the base engine. Extra arguments are currently ignored by the ggplot2 engine.
xlab	a title for the x-axis of both the Q-Q plot and the histogram: see title.
ylab	a title for the y-axis of both the Q-Q plot and the histogram: see title.
main	a title for both the Q-Q plot and the histogram: see title.
col	a colour for the bars of the histogram.
bootstrap	if TRUE then B samples will be taken from a Normal distribution with the same mean and standard deviation as x. These will be plotted in a lighter colour behind the empirical quantiles to show how much variation would be expected in the Q-Q plot for a sample of the same size from a truly normal distribution.
B	the number of bootstrap samples to take. Five should usually be sufficient.
bpch	the plotting symbol used for the bootstrap samples. Legal values are the same as any legal value for pch as defined in par.
bcol	the plotting colour used for the bootstrap samples. Legal values are the same as any legal value for col as defined in par.
shapiro.wilk	if TRUE, the P-value from the Shapiro-Wilk test for normality is displayed in the top-left corner of the Q-Q plot.

whichPlot	legal values are 1, 2, and any pair of the two, i.e. 1:2, 2:1, c(1,2), c(2,1), or variants of c(1,1). 1:2 is used by default and draws a normal Q-Q plot and a histogram of the residuals in that order. The order of the labels in xlab and ylab assume this order, and will be reordered automatically if the order is anything other than 1:2.
usePar	if TRUE, this function sets par for the user. If FALSE, this function assumes par has been set by the user and should not be overridden. Ignored by the ggplot2 engine.
engine	plotting engine to use. The default, "base", preserves the original base graphics output. Use "ggplot2" for optional ggplot2 objects.
residualType	for tslm objects, the residual scale to use in the normality plots. The default is "normalised", which checks the residuals after accounting for the fitted error correlation structure. "normalised" and "normalized" are both accepted for compatibility. Other choices are "response" and "pearson".

Details

The default base graphics engine preserves the original teaching plots and draws directly on the active graphics device. The optional ggplot2 engine is intended for users who want reusable plot objects for reports or further customisation; it requires **ggplot2** to be installed and returns ggplot objects instead of drawing base graphics side effects.

Value

Draws the selected normality diagnostic plots when using the base engine. With engine = "ggplot2", returns a ggplot object for a single selected plot or a named list of ggplot objects for multiple selected plots. When multiple ggplot2 plots are selected, printing the returned object draws the plots side by side to match the base graphics teaching layout.

See Also

shapiro.test.

Examples

```
# Synthetic teaching example: an exponential growth curve
set.seed(123)
e = rnorm(100, 0, 0.1)
x = rnorm(100)
y = exp(5 + 3 * x + e)
fit = lm(y ~ x)
normcheck(fit)

# An exponential growth curve with the correct transformation
fit = lm(log(y) ~ x)
normcheck(fit)

# Same example as above except we use normcheck.default
normcheck(residuals(fit))
```

```
# Peruvian Indians data
data(peru.df)
peruFit = lm(BP ~ weight, data = peru.df)
normcheck(peruFit)

# Optional ggplot2 engine for reusable plot objects
if (requireNamespace("ggplot2", quietly = TRUE)) {
  normPlots = normcheck(peruFit, engine = "ggplot2")
  names(normPlots)

  normcheck(peruFit, engine = "ggplot2", whichPlot = 1)
  normcheck(peruFit, engine = "ggplot2", whichPlot = 2)
}
```

`nzalc.df` *Quarterly Alcohol Available for Consumption in New Zealand*

Description

Quarterly alcohol available for consumption in New Zealand from 1935 to 2021. The data give volumes of alcoholic beverages available for consumption, grouped into broad beverage categories.

Format

A data frame with quarterly observations on 4 variables.

year Integer Year.

month Ordered factor giving the month at the end of the quarter.

volume Numeric volume available for consumption, in million litres.

category Factor beverage category: 'Total beer', 'Total wine', or 'Total spirits'.

Details

The 'month' variable gives the month ending the quarter. It should be treated in calendar order for plotting and summaries. For this quarterly data set the intended order is March, June, September, and December.

The 'category' variable has three levels:

'Total beer' Total beer available for consumption.

'Total wine' Total wine available for consumption.

'Total spirits' Total spirits and spirit-based drinks available for consumption.

Source

Stats NZ, Alcohol available for consumption: Year ended December 2021.

nzarrivals.df *Monthly Arrivals to New Zealand*

Description

Monthly international passenger arrivals to New Zealand from January 1921 to February 2026. Missing monthly observations, if present in the source series, are retained as rows with missing ‘arrivals.count‘ values.

Format

A data frame with monthly observations on 3 variables.

year Integer year.

month Factor month abbreviation with levels given by ‘month.abb‘.

arrivals.count Integer number of international passenger arrivals.

Source

Stats NZ Infoshare, table ITM049AA, Total passenger movements (monthly), Arrivals, Actual Counts. Last updated 14 April 2026.

onewayPlot *One-way Analysis of Variance Plot*

Description

Displays stripplot/boxplot of the reponse variable with intervals by factor levels. It is used as part of a one-way ANOVA analysis.

Usage

```
onewayPlot(x, ...)

## Default S3 method:
onewayPlot(
  x,
  f,
  conf.level = 0.95,
  interval.type = "tukey",
  pooled = TRUE,
  strip = TRUE,
  vert = TRUE,
  verbose = FALSE,
  ylabel = deparse(terms(formula)[[2]]),
```

```
  flabel = deparse(terms(formula)[[3]]),
  ...
)

## S3 method for class 'formula'
onewayPlot(
  formula,
  data = parent.frame(),
  conf.level = 0.95,
  interval.type = "tukey",
  pooled = TRUE,
  strip = TRUE,
  vert = TRUE,
  verbose = FALSE,
  ylabel = deparse(terms(formula)[[2]]),
  flabel = deparse(terms(formula)[[3]]),
  ...
)

## S3 method for class 'lm'
onewayPlot(x, ..., ylabel = nms[1], flabel = nms[2])
```

Arguments

x	a vector of responses, a formula object or an lm object
...	optional arguments.
f	if x is a vector of responses then f contains the group labels for each observation in x. That is, the ith value in f says which group the ith observation of x belongs to.
conf.level	confidence level of the intervals.
interval.type	three options for intervals appearing on plot: 'hsd','lsd' or 'ci'.
pooled	two options: pooled or unpooled standard deviation used for plotted intervals.
strip	if strip=F, boxplots are displayed instead.
vert	if vert=F, horizontal stripplots are displayed instead (boxplots can only be displayed vertically).
verbose	if true, print intervals on console.
ylabel	can be used to replace variable name of y by another string.
flabel	can be used to replace variable name of f by another string.
formula	a symbolic description of the model to be fit.
data	an optional data frame in which to evaluate the formula.

Methods (by class)

- `onewayPlot(default)`: One-way Analysis of Variance Plot
- `onewayPlot(formula)`: One-way Analysis of Variance Plot
- `onewayPlot(lm)`: One-way Analysis of Variance Plot

See Also

summary1way, t.test.

Examples

```
##see example in 'summary1way'

##computer data:
data(computer.df)
onewayPlot(score~selfassess, data = computer.df)


##apple data:
data(apples.df)
twosampPlot(Weight~Propagated, data = apples.df)

##oyster data:
data(oysters.df)
onewayPlot(log(Oysters)~Site, data = oysters.df)

##oyster data:
data(oysters.df)
oyster.fit = lm(log(Oysters)~Site, data = oysters.df)
onewayPlot(oyster.fit)
```

---

openCaseStudy *Open a case study source file in the editor*

Description

Opens a case study `.Rmd` file for interactive use. The file shipped inside the package is copied to `dest_dir` (so it is writable), then opened in the RStudio editor when available (otherwise the system editor).

Usage

```
openCaseStudy(id, dest_dir = getwd(), overwrite = FALSE, ...)

opencs(id, dest_dir = getwd(), overwrite = FALSE, ...)

ocs(id, dest_dir = getwd(), overwrite = FALSE, ...)
```

Arguments

`id`	Case study identifier. Flexible formats are accepted, including `"CS9_2"`, `"CS9.2"`, `"9_2"`, or `"9.2"`.

`dest_dir`	Directory to copy the case study into. Defaults to the current working directory. This legacy argument is retained for compatibility; new code may use the camelCase `destDir` alias through `....`
`overwrite`	Logical; overwrite an existing file in `dest_dir`.
`...`	Additional compatibility arguments. Currently supports `destDir`, a camelCase alias for `dest_dir`.

Value

Invisibly returns the path to the copied file.

Examples

```
if (interactive()) {
  openCaseStudy("2.1")
  openCaseStudy("2.1", destDir = tempdir())
}
```

`oysters.df` *Oyster Abundances over Different Sites*

Description

Data from an experiment to determine the abundance of oysters recruiting from three sites in two different estuaries in New South Wales. One in Georges River and two in Port Stephens. The number of oysters was recorded for 10 cm by 10 cm panels over a two year period.

Format

A data frame with 87 observations on 2 variables.

Oysters Numeric number of oysters on each experimental panel.

Site Factor giving the location of the experimental panels (GR = Georges River, PS1 = first Port Stephens site, PS2 = second Port Stephens site).

pairs20x *Pairwise Scatter Plots with Histograms and Correlations*

Description

Plots pairwise scatter plots with histograms and correlations for the data frame.

Usage

```
pairs20x(x, na.rm = TRUE, engine = c("base", "ggplot2"), ...)
```

Arguments

x	a data frame.
na.rm	if TRUE then only complete cases will be displayed.
engine	plotting engine to use. The default, "base", preserves the original base graphics output. Use "ggplot2" for the optional ggplot2/GGally output.
...	optional arguments passed to the underlying plotting function.

Details

The default base graphics engine preserves the original s20x teaching plot and draws directly on the active graphics device. The optional ggplot2 engine uses GGally when both optional packages are installed and returns a reusable plot matrix for reports or further customisation. The ggplot2/GGally output is intentionally optional so existing teaching material can continue to rely on the base graphics default.

Value

Returns the plot.

See Also

'pairs', 'panel.smooth', 'panel.cor', 'panel.hist'

Examples

```
## Peruvian Indians
data(peru.df)
pairs20x(peru.df)

# Optional ggplot2/GGally engine for a reusable plot matrix
if (requireNamespace("ggplot2", quietly = TRUE) &&
    requireNamespace("GGally", quietly = TRUE)) {
  pairsPlot = pairs20x(peru.df, engine = "ggplot2")
  class(pairsPlot)
}
```

peru.df *Peruvian Indians*

Description

A random sample of Peruvian Indians born in the Andes mountains, but who have since migrated to lower altitudes. The sample was collected to assess the long term effects of altitude on blood pressure.

Format

A data frame with 39 observations on 5 variables.

age Numeric Subject's age.

years Numeric Number of years since migration.

weight Numeric Subject's weight (kg)

height Numeric Subject's height (mm)

BP Numeric Subject's systolic blood pressure (mm Hg; standard clinical unit in New Zealand).

predict20x *Deprecated Teaching Predictions for a Linear Model*

Description

Teaching helper for linear-model predictions. It wraps `predict.lm` and prints a compact table containing fitted values, confidence intervals for the mean response, and prediction intervals for new observations.

Usage

```
predict20x(object, newdata, cilevel = 0.95, digit = 3, print.out = TRUE, ...)
```

Arguments

`object`	an `lm` object, i.e. the output from `lm`.
`newdata`	prediction data frame.
`cilevel`	confidence level for the intervals.
`digit`	number of decimal places to print.
`print.out`	if `TRUE`, print the prediction table.
`...`	optional arguments that are passed to `predict.lm`.

Details

This is not an S3 `predict()` method and is not intended to be a drop-in replacement for base R prediction methods. It is a compatibility helper for older teaching material that expects confidence and prediction intervals to be printed together. The standard `predict` interface is preferred for new work.

Note: `newdata` must be a data frame with the same column order and data types as those used in fitting the model. This is stricter than the usual `predict.lm()` interface and is kept for compatibility with the original teaching wrapper.

Value

Invisibly returns a list with components

frame printed data frame containing predictions, confidence intervals, and prediction intervals.

fit prediction values.

se.fit standard errors of predictions.

residual.scale residual standard deviation.

df residual degrees of freedom.

cilevel confidence level of the interval.

Note

This function is deprecated because it is no longer used in class. Prefer the standard `predict` method for new work.

See Also

`predict`, `predict.lm`, `as.data.frame`.

Examples

```
# Zoo data
data(zoo.df)
zoo.df = within(zoo.df, {day.type = factor(day.type)})
zoo.fit = lm(log(attendance) ~ time + sun.yesterday + nice.day + day.type + tv.ads,
             data = zoo.df)
pred.zoo = data.frame(time = 8, sun.yesterday = 10.8, nice.day = 0,
                      day.type = factor(3), tv.ads = 1.181)
predict20x(zoo.fit, pred.zoo)

# Peruvian Indians data
data(peru.df)
peru.fit = lm(BP ~ age + years + I(years^2) + weight + height, data = peru.df)
pred.peru = data.frame(age = 21, years = 2, `I(years^2)` = 2, weight = 71, height = 1629)
predict20x(peru.fit, pred.peru)
```

predictCount *Predicted Counts for a Log-Link Generalised Linear Model*

Description

Teaching helper for count predictions from a log-link generalised linear model. It wraps `predict.glm`, constructs confidence intervals on the link scale, exponentiates the fitted values and limits, rounds the result, and optionally prints the returned table.

Usage

```
predictCount(object, newdata, cilevel = 0.95, digit = 3, print.out = TRUE, ...)
```

Arguments

`object`	a `glm` object, i.e. the output from `glm`.
`newdata`	prediction data frame.
`cilevel`	confidence level for the intervals.
`digit`	number of decimal places to print.
`print.out`	if `TRUE`, print the prediction table.
`...`	optional arguments that are passed to `predict.glm`.

Details

This is not an S3 `predict()` method and is not intended to be a drop-in replacement for base R prediction methods. It is a specialised count-focused teaching wrapper. For a more general log-link or logit-link GLM helper, see `predictGLM`.

Note: `newdata` must be a data frame with the same column order and data types as those used in fitting the model. This stricter interface is kept for compatibility with the original teaching wrapper.

Value

Invisibly returns a data frame with three columns:

Predicted the predicted count on the response scale.

Conf.lower the lower confidence limit on the response scale.

Conf.upper the upper confidence limit on the response scale.

See Also

`predict`, `predict.glm`, `predictGLM`, `as.data.frame`.

predictGLM *Prediction Intervals for Log-Link and Logit-Link Generalised Linear Models*

Description

Teaching helper for predictions from log-link and logit-link generalised linear models. It wraps predict.glm with standard errors and returns fitted values with confidence limits on either the link scale or the response scale.

Usage

```
predictGLM(object, newdata, type = "link", cilevel = 0.95, quasit = FALSE, ...)
```

Arguments

object	a glm object, i.e. the output from glm.
newdata	prediction data frame.
type	"link" (default) or "response" for estimates and confidence intervals on the linear predictor or response scale.
cilevel	confidence level for the intervals.
quasit	if TRUE, use a t multiplier rather than a normal multiplier for confidence intervals when object is a quasi model.
...	optional arguments that are passed to predict.glm.

Details

This is not an S3 predict() method and is not intended to be a drop-in replacement for base R prediction methods. It is the more general GLM teaching helper in this package; predictCount remains a specialised count-focused wrapper with rounded response-scale output.

Note: newdata must include all first-order terms used in the fitted model. This simplified requirement reflects the teaching-wrapper interface and is not a complete reproduction of predict.glm().

Value

A data frame with columns fit, lwr, and upr containing fitted values and confidence limits on the requested scale.

See Also

predict, predict.glm, predictCount.

print.s20xModelcheck_ggplot2

Print ggplot2 modelcheck plots

Description

Draws multiple ggplot2 modelcheck plots together so the optional ggplot2 engine gives a single printed diagnostic display rather than showing list structure at the console.

Usage

```
## S3 method for class 's20xModelcheck_ggplot2'
print(x, ...)
```

Arguments

x	an object returned by modelcheck(..., engine = "ggplot2") when multiple plots are selected.
...	additional arguments passed to print.ggplot.

Value

Invisibly returns x.

print.s20xNormcheck_ggplot2

Print ggplot2 normcheck plots

Description

Draws multiple ggplot2 normcheck plots side by side so the optional ggplot2 engine mirrors the base graphics layout for the default whichPlot = 1:2 case.

Usage

```
## S3 method for class 's20xNormcheck_ggplot2'
print(x, ...)
```

Arguments

x	an object returned by normcheck(..., engine = "ggplot2") when multiple plots are selected.
...	additional arguments passed to print.ggplot.

Value

Invisibly returns x.

propslsd.new *LSD-Display Intervals*

Description

This function is called by rowdistr.

Usage

```
propslsd.new(crosstablist, conf.level = 0.95, arrowlength = 0.1)
```

Arguments

crosstablist	A list produced by crosstabs or a matrix containing a 2-way table of counts (without marginal totals).
conf.level	Confidence level of the intervals.
arrowlength	Length of the arrows.

Note

This is an internal legacy helper used by rowdistr(). It is not exported and should not be called directly by users.

See Also

crosstabs, rowdistr

rain.df *Cloud Seeding and Levels of Rainfall*

Description

Data from an experiment to see if seeding clouds with Silver Nitrate effects the amount of rainfall.

Format

A data frame with 50 observations on 3 variables.

rain Numeric amount of rain, measured in acre-feet (the volume of water required to cover one acre of land to a depth of one foot).

rain_ML Numeric amount of rain expressed in megalitres (ML); rain_ML = rain * 1.23348184.

seed Factor indicating whether the clouds were seeded (seeded, unseeded).

`residPlot` *Fitted values versus residuals plot*

Description

Plots a scatter plot for the variables of the residuals and fitted values from the linear model, lmfit. A lowess smooth line for the underlying trend, as well as one standard deviation error bounds for the scatter about this trend, are added to this scatter plot. A test for a quadratic relationship between the residuals and the fitted values is also computed.

Usage

```
residPlot(lmfit, f = 0.5)
```

Arguments

`lmfit`	an `lm` object, i.e. the output from `lm`.
`f`	the smoother span. This gives the proportion of points in the plot which influence the smooth at each value. Larger values give more smoothness.

Value

Returns the plot.

Note

This is a legacy diagnostic plotting helper retained for compatibility with older teaching material. New code should usually prefer the current diagnostic workflow used by `modelcheck()`.

See Also

`trendscatter`

Examples

```
# Peruvian Indians data
data(peru.df)
fit = lm(BP ~ age + years + weight + height, data = peru.df)
residPlot(fit)
```

rowdistr *Row distributions from a cross-tabulation of two variables*

Description

Produces summaries and plots from a cross-tabulation. The output produced depends on the parameter 'comp'. Columns relate to response categories and rows to different populations.

Usage

```
rowdistr(
  crosstablist,
  comp = c("basic", "within", "between"),
  conf.level = 0.95,
  plot = TRUE,
  suppressText = FALSE
)
```

Arguments

crosstablist	a list produced by 'crosstabs' or a matrix containing a 2-way table of counts (without marginal totals).
comp	three options: 'basic' (default), 'within', and 'between'.
conf.level	confidence level of the intervals.
plot	if FALSE then the row distribution plots are not displayed
suppressText	if TRUE then text results are not displayed

Details

The 'basic' option (default) produces the response distribution for each row population together with comparative bar charts.

If comp = 'between' the resulting output displays how the probability of falling into a response class (column) differs between populations. Confidence intervals for differences in proportions are produced together with a set of barcharts with LSD intervals.

If comp = 'within' the resulting output shows the extent to which the component probabilities of the same row distribution differ. Separate Chi-square tests for uniformity are produced for each row distribution as are confidence intervals for differences in proportions within the same distribution.

Arguments plot and suppressText are really only used when producing knitr or Sweave documents so that just the plot or just the text can be displayed in the document.

Value

Invisibly returns the matrix of row proportions printed by the teaching summary when suppressText = FALSE. When suppressText = TRUE, the function invisibly returns NULL because no text summary is constructed. Plotting remains a side effect controlled by plot.

See Also

crosstabs

Examples

```
data(body.df)
z = crosstabs(~ ethnicity + married, data = body.df)
rowdistr(z)
rowdistr(z, comp = "between")
rowdistr(z, comp = "within")

## from matrix of counts
z = matrix(c(4, 3, 2, 6, 47, 20, 40, 62, 11, 8, 7, 22, 3, 0, 1, 10), 4, 4)
rowdistr(z)
```

---

rr *Read Data*

Description

For internal use.

Usage

```
rr()
```

---

seeds.df *Seeds Data*

Description

These data record the number of seeds (out of 100) that germinated when given different amounts of water. The seeds were either exposed to light or kept in the dark. Four identical boxes were used for each combination of water and light

Format

A data frame with 48 observations on 3 variables.

Light Factor indicating whether the seeds were exposed to light (N = No, Y = Yes).

Water Integer amount of water, with higher levels corresponding to more water (1, 2, 3, 4, 5, 6).

Count Integer number of seeds that germinated, out of 100.

sheep.df *Sheep Data*

Description

Weight measurements for sheep under combinations of copper and cobalt supplementation.

Format

A data frame with 100 observations on 3 variables.

Weight Integer Weight of sheep (kilograms, kg).

Copper Factor indicating whether copper supplementation was given (No, Yes).

Cobalt Factor indicating whether cobalt supplementation was given (No, Yes).

skewness *Skewness Statistic*

Description

Calculates the skewness statistic of the data in 'x'. Values close to zero correspond to reasonably symmetric data, positive values of this measure indicate right-skewed data whereas negative values indicate left-skewness.

Usage

```
skewness(x, ...)
```

Arguments

x	vector containing the data.
...	any other variables to be passed to mean and sd, e.g. na.rm = TRUE.

Value

Returns the value of the skewness.

Examples

```
## Merger data:
data(mergers.df)
skewness(mergers.df$mergerdays)
```

skulls.df *Skulls Data*

Description

Male Egyptian skulls from five different epochs. Each skull has had four measurements taken of it, BH, Basibregmatic Height, BL, Basialveolar Length, MB, Maximum Breadth and NH, Nasal Height. It is of interest to investigate the change in shape over time. A gradual change, would indicate inbreeding of the populations. This data only includes the maximum breadth measurements.

Format

A data frame with 150 observations on 2 variables.

measurement Integer maximum breadth measurement of the skull.

year Integer epoch year group for the skull.

Source

A Handbook of Small Data Sets

References

Hand, D.J., Daly, F., Lunn, A.D., McConway, K.J. and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. Boca Raton, Florida: Chapman and Hall/CRC.

Thomson, A. and Randall-Maciver, R. (1905). *Ancient Races of the Thebaid*. Oxford: Oxford University Press.

snapper.df *Snapper Weight Data*

Description

Weight and length measurements of 844 snapper (Pagrus auratus) caught in the Hauraki Gulf, near Auckland, New Zealand.

Format

A data frame with 844 observations on 2 variables.

len Numeric fork length in centimetres. Fork length is measured from the tip of the snout to the fork of the tail.

wgt Numeric weight of the fish, in kilograms.

Source

Russell Millar, University of Auckland.

soyabean.df *Soya Bean Yields*

Description

Data from an experiment to examine the effects of different planting times on the yield of soya beans, given four different cultivars.

Format

A data frame with 32 observations on 3 variables.

yield Numeric Yield of each plant.

cultivar Factor Cultivar used (cult1, cult2, cult3, cult4)

planttime Factor Month of planting (Novemb, Decemb)

Source

Littler, R. University of Waikato

stripqq *Deprecated strip charts and normal quantile-quantile plots*

Description

‘stripqq()‘ is deprecated and is no longer exported. It draws strip charts and normal quantile-quantile plots of ‘x‘ for each level of the grouping variable ‘g‘.

Usage

```
stripqq(formula, ...)
```

Arguments

formula	A symbolic specification of the form x ~ g can be given, indicating the observations in the vector x are grouped according to the levels of the factor g. NAs are allowed in the data.
...	Optional arguments that are passed to the stripchart function.

Note

This is a legacy teaching helper retained for compatibility with older course material. New teaching material should prefer current diagnostic plotting workflows.

summary1way *One-way Analysis of Variance Summary*

Description

Displays summary information for a one-way anova analysis. The lm object must come from a numerical response variable and a single factor. The output includes: (i) anova table; (ii) numeric summary; (iii) table of effects; (iv) plot of data with intervals.

Usage

```
summary1way(
  fit,
  digit = 5,
  conf.level = 0.95,
  inttype = "tukey",
  pooled = TRUE,
  print.out = TRUE,
  draw.plot = TRUE,
  ...
)
```

Arguments

fit	an lm object, i.e. the output from lm.
digit	decimal numbers after the point.
conf.level	confidence level of the intervals.
inttype	three options for intervals appeared on plot: 'hsd','lsd' or 'ci'.
pooled	two options: pooled or unpooled standard deviation used for plotted intervals.
print.out	if TRUE, print out the output on the screen.
draw.plot	if TRUE, plot data with intervals.
...	more options.

Value

Invisibly returns a list containing the one-way ANOVA summary components used in the printed teaching output. The list contains:

Df	degrees of freedom for between groups, within groups, and total.
Sum of Sq	sum of squares for between groups, within groups, and total.
Mean Sq	mean squares for between groups and within groups.
F value	the one-way ANOVA F statistic.
Pr(F)	the P-value associated with the F test.
Main Effect	the grand mean of the response.

`Group Effects` group deviations from the grand mean.

The printed ANOVA table, numeric summary, effects table, and optional plot are the primary teaching interface. The returned list is invisible so classroom use can focus on the printed output while programmatic callers can still inspect the computed values.

See Also

summary2way, anova, aov, dummy.coef, onewayPlot

Examples

```
## Computer questionnaire data:
data(computer.df)
computer.df = within(computer.df, {
    selfassess = factor(selfassess)
})
computer.fit = lm(score ~ selfassess, data = computer.df)
result = summary1way(computer.fit)
result
```

---

summary2way *Two-way Analysis of Variance Summary*

Description

Displays summary information for a two-way anova analysis. The lm object must come from a numerical response variable and factors. The output depends on the value of page:

Usage

```
summary2way(
  fit,
  page = c("table", "means", "effects", "interaction", "nointeraction"),
  digit = 5,
  conf.level = 0.95,
  print.out = TRUE,
  new = TRUE,
  all = FALSE,
  FUN = "identity",
  ...
)
```

Arguments

fit	an lm object, i.e. the output from lm().
page	options for output: "table", "means", "effects", "interaction", or "nointeraction".
digit	the number of decimal places in the display.
conf.level	confidence level of the intervals.
print.out	if TRUE, print the output on the screen.
new	if TRUE then this will run the new version of summary2way which should be more robust than the old version. However, it does not work in the same way. In particular, when page = 'means' it does not return summary statistics for each grouping of the data (pooled, by row factor, by column factor, and by interaction factor). Instead, it simply returns the means for each grouping.
all	Only applicable to page = "interaction". If TRUE, pairwise comparisons for all combinations of factor levels are shown. Otherwise, comparisons are only shown between combinations that have the same level for one of the factors.
FUN	optional function to be applied to estimates and confidence intervals. Typically for backtransformation operations.
...	other arguments such as inttype and pooled.

Details

- page = "table": ANOVA table.
- page = "means": cell means matrix and numeric summary.
- page = "effects": table of effects.
- page = "interaction": interaction contrast tables.
- page = "nointeraction": main-effect contrast tables.

Value

'summary2way()' prints the requested teaching summary page and invisibly returns the current summary components. The returned list has the following components:

Df	degrees of freedom for regression, residual and total.
Sum of Sq	sum squares for regression, residual and total.
Mean Sq	mean squares for regression and residual.
F value	F-statistic value.
Pr(F)	The P-value associated with each F-test.
Grand Mean	The overall mean of the response variable.
Row Effects	The main effects for the first (row) factor.
Col Effects	The main effects for the second (column) factor.
Interaction Effects	The interaction effects if an interaction model has been fitted, otherwise NULL.

`results` If `new = TRUE`, then this is a list with five components: `table` - the ANOVA table, `means` the table of means from `model.tables`, `effects` - the table of effects from `model.tables`, and `comparisons` - the differences in the means with standard errors, confidence bounds, and P-values from `TukeyHSD`

.

See Also

`summary1way`, `model.tables`, `TukeyHSD`

Examples

```
## Arousal data:
data(arousal.df)
arousal.fit = lm(arousal ~ gender * picture, data = arousal.df)
summary2way(arousal.fit)

## Butterfat data:
data("butterfat.df")
fit = lm(log(Butterfat) ~ Breed + Age, data = butterfat.df)
summary2way(fit, page = "nointeraction", FUN = exp)
```

---

summaryStats *Summary Statistics*

Description

Produces a table of summary statistics for the data. If the argument `group` is missing, calculates a matrix of summary statistics for the data in `x`. If `group` is present, the elements of `group` are interpreted as group labels and the summary statistics are displayed for each group separately.

Usage

```
summaryStats(x, ...)

## Default S3 method:
summaryStats(
  x,
  group = rep("Data", length(x)),
  data.order = TRUE,
  digits = 2,
  ...
)

## S3 method for class 'formula'
summaryStats(x, data = NULL, data.order = TRUE, digits = 2, ...)
```

```
## S3 method for class 'matrix'
summaryStats(x, data.order = TRUE, digits = 2, ...)
```

Arguments

x	either a single vector of values, a formula of the form data ~ group, or a matrix.
...	Optional arguments that are passed to the summary statistic functions. For example na.rm = TRUE will help if there are missing values in the (response) variable.
group	a vector of group labels.
data.order	if TRUE, the group order is the order in which the groups are first encountered in group. If FALSE, the order is alphabetical.
digits	the number of decimal places to display.
data	an optional data frame containing the variables in the model.

Value

A teaching summary is printed as a side effect. The returned value is invisible so that classroom use can focus on the printed summary while programmatic use can still save the result.

If x is a single variable and no grouping is supplied, an invisible list is returned with the following named items:

min	Minimum value.
max	Maximum value.
mean	Mean value.
var	Variance – the average of the squares of the deviations of the data values from the sample mean.
sd	Standard deviation – the square root of the variance.
n	Number of data values – size of the dataset.
nMissing	If there are missing values, and na.rm has been set to TRUE, the number of missing values.
iqr	Midspread (IQR) – the range spanned by the central half of the data; the interquartile range.
skewness	Skewness statistic – indicates how skewed the data set is. Positive values indicate right-skew data. Negative values indicate left-skew data.
lq	Lower quartile.
median	Median – the middle value when the batch is ordered.
uq	Upper quartile.

If grouping is provided, either by using the group argument, by using a formula, or by passing a matrix whose columns represent groups, the function invisibly returns a data.frame with one row for each group and columns containing the summary statistics.

Methods (by class)

- summaryStats(default): Summary Statistics
- summaryStats(formula): Summary Statistics
- summaryStats(matrix): Summary Statistics

Examples

```
## STATS20x data:
data(course.df)

## Single variable summary
with(course.df, summaryStats(Exam))

## Using a formula
summaryStats(Exam ~ Stage1, course.df)

## Using a matrix
courseMatrix = cbind(course.df$Exam, course.df$Assign, course.df$Test)
summaryStats(courseMatrix)

## Saving and extracting the information
sumStats = summaryStats(Exam ~ Degree, course.df)
sumStats

## Just the BAs
sumStats['BA', ]

## Just the means
sumStats$mean
```

---

teach.df *Comparison of Three Teaching Methods*

Description

Data from an experiment to assess the impact of three different teaching methods on language ability. 30 students were randomly allocated into three groups, one for each method. The students' IQ before instruction and a language test score after instruction were recorded.

Format

A data frame with 30 observations on 3 variables.

lang Numeric Language test score after instruction.

IQ Numeric Student's IQ.

method Factor Teaching method (1, 2, 3)

technitron.df *Technitron Salary Information*

Description

Salary information for all salaried employees of the Technitron Company.

Format

A data frame with 46 observations on 8 variables.

salary Numeric Annual Salary (dollars)

yrs.empl Numeric Number of years employed at Technitron.

prior.yrs Numeric Number of years prior experience.

educ Numeric Years of education after high school.

id Numeric Company identification number.

gender Numeric Gender (0 = female, 1 = male)

dept Numeric Department employee works in (1 = Sales, 2 = Purchasing, 3 = Advertising, 4 = Engineering)

super Numeric Number of employees supervised.

thyroid.df *Effect of a New Drug on Thyroid Weights*

Description

Data from an experiment to assess the effect of a new drug on the weight of the thyroid gland using 16 laboratory animals. The animals were randomly assigned into either a control group, or a treatment group, and each animal had its bodyweight recorded at the beginning of the experiment and its thyroid weight measured at the end of the experiment.

Format

A data frame with 16 observations on 3 variables.

thyroid Numeric Weight of thyroid gland after 7 days (mg)

body Numeric Animal body weight before experiment began (g)

group Factor Animal's group (1 = control, 2 = drug)

toothpaste.df *Crest Toothpaste*

Description

Two random samples of households, one of households who purchase Crest toothpaste and one of households who do not. For each household the age is recorded of the person responsible for purchasing the toothpaste.

Format

A data frame with 20 observations on 2 variables.

purchasers Numeric Age of the person in the household responsible for purchases of Crest.

nonpurchasers Numeric Age of the person in the household responsible for purchases of other brands of toothpaste.

trendscatter *Trend and scatter plot*

Description

Plots a scatter plot for the variables x, y along with a lowess smooth for the underlying trend. One standard deviation error bounds for the scatter about this trend are also plotted.

Usage

```
trendscatter(x, ...)

## Default S3 method:
trendscatter(x, y = NULL, f = 0.5, xlab = NULL, ylab = NULL, main = NULL, ...)

## S3 method for class 'formula'
trendscatter(
  x,
  f = 0.5,
  data = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  ...
)
```

Arguments

x	the coordinates of the points in the scatter plot. Alternatively, a formula.
...	Optional arguments
y	the y coordinates of the points in the plot, ignored if x is a function.
f	the smoother span. This gives the proportion of points in the plot which influence the smooth at each value. Larger values give more smoothness.
xlab	a title for the x axis: see title.
ylab	a title for the y axis: see title.
main	a title for the plot: see title.
data	an optional data frame containing the variables in the model.

Value

Returns the plot.

Methods (by class)

- trendscatter(default): Trend and scatter plot
- trendscatter(formula): Trend and scatter plot

See Also

residPlot

Examples

```
# Synthetic teaching example: a simple polynomial
set.seed(123)
x = rnorm(100)
e = rnorm(100)
y = 2 + 3 * x - 2 * x^2 + 4 * x^3 + e
trendscatter(y ~ x)

# Synthetic teaching example: an exponential growth curve
e = rnorm(100, 0, 0.1)
y = exp(5 + 3 * x + e)
trendscatter(log(y) ~ x)

# Peruvian Indians data
data(peru.df)
trendscatter(BP ~ weight, data = peru.df)

# Note: this usage is deprecated
with(peru.df, trendscatter(weight, BP))
```

`tslm` *Fit a linear model with optional autoregressive errors*

Description

'tslm()' is a teaching-friendly wrapper for fitting linear models with optional AR(p) error structures. Students specify the mean model using an ordinary formula and add an 'ar(p)' term to request autoregressive errors.

Usage

```
tslm(formula, data = parent.frame(), time, method = "REML", ...)
```

Arguments

`formula`	a model formula. Use 'ar(p)' in the right hand side to specify AR(p) errors, for example 'y ~ x + ar(1)'.
`data`	an optional data frame containing the variables in the model. If omitted, variables are taken from the calling environment.
`time`	optional unquoted or quoted name of the time variable in 'data' or in the calling environment. If omitted for an AR model, the row order of the model data is used.
`method`	fitting method passed to [nlme::gls()] for AR models. Defaults to '"REML"'.
`...`	additional arguments passed to [stats::lm()] or [nlme::gls()].

Details

When no 'ar(p)' term is present, 'tslm()' fits an ordinary [stats::lm()] model. When an 'ar(p)' term is present, 'tslm()' fits a [nlme::gls()] model with an AR(p) correlation structure using [nlme::corARMA()]. The 'ar(p)' term changes the error model, not the mean-model terms printed in the formula.

The formula describes the mean model, just as it does for [stats::lm()]. The special term 'ar(p)' is removed from the mean model before fitting and is used only to specify the correlation structure for the errors. For example, 'log(passengers) ~ t + month + ar(1)' fits a trend and seasonal mean model with AR(1) errors.

For AR-error models, 'time' should usually name the variable giving the time order of the observations. If 'time' is omitted, 'tslm()' fits the model using the row order of the model data and gives a warning so that this assumption is visible.

Diagnostic methods for AR-error models use normalised residuals by default, because these residuals account for the fitted correlation structure. Use 'residualType = "response"' when the raw response residuals are required. '"normalised"' and '"normalized"' are both accepted for compatibility.

Value

An object of class 'tslm', containing the original formula, the mean formula fitted internally, the AR order, the time variable if supplied, and the underlying fitted model.

See Also

[stats::lm()], [nlme::gls()], [nlme::corARMA()]

Examples

```
data(beer.df)
fit = tslm(beer ~ t + ar(1), data = beer.df, time = t)
coef(fit)

data(airpass.df)
fitAr = tslm(log(passengers) ~ t + month + ar(1),
  data = airpass.df,
  time = t
)
summary(fitAr)
anova(fitAr)

plot(fitAr)
plot(fitAr, residualType = "response")
```

---

zoo.df *Zoo Attendance during an Advertising Campaign*

Description

Data for 455 days of attendance records for Auckland Zoo, from January 1, 1993. Note that only 440 values are given due to missing values. It was of interest to assess whether an advertising campaign was effective in increasing attendance.

Format

A data frame with 440 observations on 6 variables.

attendance Numeric Number of visitors.

time Numeric Time in days since the start of the study.

sun.yesterday Numeric Hours of sunshine the previous day.

tv.ads Numeric Average spending on TV advertising in the previous week (1000s of dollars per day)

nice.day Factor Assessment based on number of hours of sunshine (0 = No, 1 = Yes)

day.type Factor Type of day (1 = ordinary weekday, 2 = weekend day, 3 = school holiday weekday, 4 = public holiday)

Index