Package ‘SciencesPo’

January 22, 2016

Type Package

Title A Tool Set for Analyzing Political Behavior Data

Date/Publication 2013-08-27

Version 1.3.9

Date 2016-01-20

Maintainer Daniel Marcelino <dmarcelino@live.com>

Description Provides functions for analyzing political behavior data, including measures of political fragmentation, seat apportionment, and graphical demonstrations.

URL http://CRAN.R-project.org/package=SciencesPo

License GPL (>= 2)

Depends R (>= 3.2.0),
ggplot2 (>= 2.0.0),
stats,
utils,
graphics,
grDevices,
methods

Imports RSQLite (>= 1.0.0),
data.table (>= 1.9.4),
grid (>= 3.0.0),
gridExtra,
magrittr,
stringr,
shiny,
lazyeval,
dplyr,
vcd

Suggests testthat,
scales,
knitr

VignetteBuilder knitr

LazyData yes

Repository CRAN

BugReports http://github.com/danielmarcelino/SciencesPo/issues

ByteCompile TRUE

RoxygenNote 5.0.1

R topics documented:

aad *Average Absolute Deviation*

Description

Calculates the average (mean) absolute deviation from the sample mean.

Usage

```
aad(x, na.rm = TRUE, ...)
```

Arguments

`x`	A numeric vector containing the observations.
`na.rm`	A logical value for `na.rm`, default is `na.rm=TRUE`.
`...`	Additional arguements (currently ignored)

Details

The statistical literature has not yet adopted a standard notation for the "Mean Absolute Deviation" and the "Median Absolute Deviation". As a result, both statistics have been denoted as "MAD", which may lead to confusion once they may produce different values. The R `mad` by default computes the "Median Absolute Deviation"; to obtain the "Mean Absolute Deviation" one has to use `mad(x, constant = 1)`. Thus, the function `aad` will calculate the "Mean Absolute Deviation"–or "Average Deviation (AD)" as proposed by Garrett, who defines it as "the mean of the deviation of all the separate scores in the series taken from their mean (occasionally from the median or mode)", (1971, p. 481).

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Garrett, Henry (1982) *Statistics in Psychology and Education*. 6th, Paragon.

See Also

`mad`

Examples

```
x <- c(15, 10, 6, 8, 11)
aad(x)
```

aad-class *An S4 Class to Average Absolute Deviation*

Description

An S4 Class to Average Absolute Deviation

Slots

`estimate` Estimated value.

agostino *D'Agostino test of skewness*

Description

Performs the D'Agostino test for skewness in normally distributed data.

Usage

```
agostino(x, alternative = c("two.sided", "less", "greater"))
```

Arguments

`x`	A numeric vector of data values.
`alternative`	A character string specifying the alternative hypothesis, must be one of '"two.sided"' (default), '"greater"' or '"less"'. You can specify just the initial letter.

Details

Under the hypothesis of normality, data should be symmetrical (i.e. skewness should be equal to zero). This test has such null hypothesis and is useful to detect a significant skewness in normally distributed data.

References

D'Agostino, R.B. (1970). Transformation to Normality of the Null Distribution of G1. Biometrika, 57, 3, 679-681.

Examples

```
set.seed(1234)
x = rnorm(1000)

skewness(x) # is data normal?

agostino(x)
```

`anderson.darling` *Anderson-Darling test for normality*

Description

Performs the Anderson-Darling test for the composite hypothesis of normality. See details.

Usage

```
anderson.darling(x)
```

Arguments

`x`	A numeric vector of data values, the number of observations must be greater than 7. Missing values are allowed.

Details

The Anderson-Darling test is the recommended EDF test by Stephens (1986) followed by the Cramer-von Mises test. Compared to the later, the Anderson-Darling gives more weight to the tails of the distribution.

Examples

```
set.seed(1234)
x = rnorm(1000)
anderson.darling(x)
```

`anscombe.glynn` *Anscombe-Glynn test of kurtosis*

Description

Performs the Anscombe-Glynn test of kurtosis for normal samples.

Usage

```
anscombe.glynn(x, alternative = c("two.sided", "less", "greater"))
```

Arguments

`x`	A numeric vector of data values.
`alternative`	A character string specifying the alternative hypothesis, must be one of '"two.sided"' (default), '"greater"' or '"less"'. You can specify just the initial letter.

Details

Under the hypothesis of normality, data should have kurtosis equal to 3.This test has such null hypothesis and is useful to detect a significant difference of kurtosis in normally distributed data.

References

Anscombe, F.J., Glynn, W.J. (1983) Distribution of kurtosis statistic for normal statistics. Biometrika, 70, 1, 227-234

Examples

```
set.seed(1234)
x = rnorm(1000)
kurtosis(x)
anscombe.glynn(x)
```

`atkinson` *Atkinson Index of Inequality*

Description

Calculates the Atkinson Index. This inequality measure is espcially good at determining which end of the distribution is contributing most to the observed inequality.

Usage

```
atkinson(x, n = rep(1, length(x)), parameter = 0.5, na.rm = FALSE, ...)

atkinson(x, n = rep(1, length(x)), parameter = 0.5, na.rm = FALSE, ...)
```

Arguments

`x`	A vector of data values of non-negative elements.
`n`	A vector of frequencies of the same length as `x`.
`parameter`	A parameter of the inequality measure (if set to `NULL` the default parameter of the respective measure is used).
`na.rm`	A logical. Should missing values be removed? The Default is set to `na.rm=FALSE`.
`...`	Additional arguements (currently ignored)

References

Cowell, F. A. (2000) Measurement of Inequality in Atkinson, A. B. / Bourguignon, F. (Eds): *Handbook of Income Distribution*. Amsterdam.

Cowell, F. A. (1995) *Measuring Inequality* Harvester Wheatshef: Prentice Hall.

See Also

`herfindahl`, `rosenbluth`, `gini`. For more details see the Indices vignette: `vignette("Indices", package = "Scienc`

Examples

```
if (interactive()) {
# generate a vector (of incomes)
x <- c(778, 815, 857, 888, 925, 930, 965, 990, 1012)

# compute Atkinson coefficient with parameter=0.5
atkinson(x, parameter=0.5)
}
```

bar.plot — *Plot a barplot with ggplot2*

Description

Plot a barplot with ggplot2.

Usage

```
bar.plot(data, x.var = NULL, y.var = NULL, group.var = NULL,
  group.colors = NULL, palette = NULL, stat = "identity", ...)
```

Arguments

`data`	The data frame.
`x.var`	The name of column containing x variable. Default value is NULL.
`y.var`	The name of column containing y variable.
`group.var`	The name of column containing group variable. This variable is used to color plot according to the group.
`group.colors`	The color of groups; group.colors should have the same length as groups.
`palette`	This can be also used to indicate group colors. In this case the parameter group.colors should be NULL.
`stat`	The statistical transformation to use on the data for this layer; default value is `identity`. To get a bar graph of counts, don't map a variable to y, and use `stat="bin"`
`...`	Other parameters passed on to ggplot2.customize function.

Examples

```
if (interactive()) {
x = sample(10, 100, rep = TRUE)
y = stats::rnorm(100)
z = sample(letters[1:3],100, rep=TRUE)
dat = data.frame(x,y,z)

bar.plot(dat, 'x', 'y')
bar.plot(dat, 'x', 'y', group.var = 'z')
}
```

bartels.rank — *Bartels Rank Test of Randomness*

Description

Performs Bartels rank test of randomness. The default method for testing the null hypothesis of randomness is `two.sided`. By using the alternative `left.sided`, the null hypothesis is tested against a trend. By using the alternative `right.sided`, the null hypothesis of randomness is tested against a systematic oscillation in the observed data.

Usage

```
bartels.rank(x, alternative = "two.sided", pvalue = "normal")

## Default S3 method:
bartels.rank(x, alternative = "two.sided",
  pvalue = "normal")
```

Arguments

`x`	A numeric vector of data values.
`alternative`	A method for hypothesis testing, must be one of "two.sided" (default), "left.sided" or "right.sided".
`pvalue`	A method for asymptotic aproximation used to compute the p-value.

Details

Missing values are by default removed. The RVN test statistic is

$$RVN = \frac{\sum_{i=1}^{n-1}(R_i - R_{i+1})^2}{\sum_{i=1}^{n}\left(R_i - (n+1)/2\right)^2}$$

where $R_i = rank(X_i), i = 1, \dots, n$. It is known that $(RVN - 2)/\sigma$ is asymptotically standard normal, where $\sigma^2 = \frac{4(n-2)(5n^2-2n-9)}{5n(n+1)(n-1)^2}$.

Value

`statistic`	The value of the RVN statistic test and the theoretical mean value and variance of the RVN statistic test.
`n`	the sample size, after the remotion of consecutive duplicate values.
`p.value`	the asymptotic p-value.
`method`	a character string indicating the test performed.
`data.name`	a character string giving the name of the data.
`alternative`	a character string describing the alternative.

References

Bartels, R. (1982). The Rank Version of von Neumann's Ratio Test for Randomness, *Journal of the American Statistical Association*, **77**(377), 40-46.

Gibbons, J.D. and Chakraborti, S. (2003). *Nonparametric Statistical Inference*, 4th ed. (pp. 97-98). URL: http://books.google.pt/books?id=dPhtioXwI9cC&lpg=PA97&ots=ZGaQCmuEUq

Examples

```
# Example 5.1 in Gibbons and Chakraborti (2003), p.98.
# Annual data on total number of tourists to the United States for 1970-1982.
 years <- 1970:1982
tourists <- c(12362, 12739, 13057, 13955, 14123,  15698, 17523,
 18610, 19842, 20310, 22500, 23080, 21916)

 # See it graphically
 qplot(factor(years), tourists)+ geom_point()
```

```
# Test the null against a trend
 bartels.rank(tourists, alternative="left.sided", pvalue="beta")
```

---

bhodrick93 *Bekaert's and Hodrick's (1993) Data*

Description

Data set used by Bekaert and Hodrick (1993) on biases in the measurement of foreign exchange risk premiums. This dataset contains the following columns:

- date A character vector for date.
- jyspot Price of USD in JY, spot.
- jyfwd Price of USD in JY, 30-day forward.
- jys30 Price of USD in JY, spot market at 30-day forward deliver/date.
- dmspot Price of USD in DM, spot.
- dmfwd Price of USD in DM, 30-day forward
- dms30 Price of USD in DM, spot market at 30-day forward deliver/date.
- bpspot Price of USD in BP, spot.
- bpfwd Price of USD in BP, 30-day forward.
- bps30 Price of USD in BP, spot market at 30-day forward deliver/date.
- quote A numeric vector.

Usage

```
data(bhodrick93)
```

Format

A `data.frame` object with 11 variables and 778 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Bekaert, G., and Hodrick, R. J. (1993) On biases in the measurement of foreign exchange risk premiums. *Journal of International Money and Finance,* **12(2),** 115-138.

binomcdf — *Binomial cumulative distribution function*

Description

Computes a binomial cdf at each of the values in x using the corresponding number of trials in n and probability of success for each trial in p.

Usage

```
binomcdf(n, p, x)
```

Arguments

n	the number of trials.
p	a vector of probabilities.
x	the number of success.

Examples

```
trials = 10
prob = c(.2,.25,.3,.35)
success = 4
binompdf(n = trials, p = prob, x = success)
```

binompdf — *Binomial probability density function*

Description

Computes the binomial pdf at each of the values in x using the corresponding number of trials in n and probability of success for each trial in p.

Usage

```
binompdf(n, p, x)
```

Arguments

n	the number of trials.
p	a vector of probabilities.
x	the number of success.

Note

The probability density function (pdf) is given by:

$$p(x) = \binom{n}{k} p^x (1-p)^{n-x}$$

with $x = 0, 1, 2, \ldots$

Examples

```
trials = 10
prob = c(.2,.25,.3,.35)
success = 4
binomcdf(n = trials, p = prob, x = success)
```

---

blockRandomizedDesign *Create Block-randomized designs*

Description

Generate block-randomized designs based on the number of units n and block size, where the block size is the number of experimental conditions. The number of Independent Variables and the number of levels in each IV are specified as input. The output is a the block randomized design. This function is intended for planning randomized trails.

Usage

```
blockRandomizedDesign(blocksize, n, seed = NULL)
```

Arguments

blocksize	is the number of control blocks or n per block/group.
n	is the total number of subjects or units.
seed	the random number generation seed.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

blk <- blockRandomizedDesign(blocksize = 20, n = 80, seed = 51) blk; table(blk$block, blk$condition)

let's do some analysis set.seed(51); blk$y <- rnorm(n = 80, mean = 20, sd = 5)

Let's look at some descriptives: tapply(blk$y, list(blk$condition, blk$block), mean) tapply(blk$y, list(blk$condition, blk$block), sd)

Do the ANOVA and make some graphs # This formula describes the response 'y' by both the treatment factor 'condition' and the block control 'block'. Note that aov() treats 'block' as a random error component of the variance, while lm() treats 'block' as a fixed effect.

fit.aov <- aov(y ~ factor(condition) + factor(block), data=blk) summary(fit.aov) # display Type I ANOVA table drop1(fit.aov,~.,test="F") # type III SS and F Tests

Since the p-value of 0.254 is much greater than the .05 significance level, we cannot reject the null hypothesis that the mean of 'y' for each treatment conditions are all equal.

model.tables(fit.aov, "means", se=TRUE) # SE for differences, NOT for means # Calculate the pooled standard error of the means. pooled.se = sqrt(1688.1/4)

block <- c(1,2,3,4) # the values of the x axis outcome <- c(19.76, 20.03, 18.44, 18.16) # the results from the means output plot(block, outcome, type = "b", ylab = "outcome", xlab = "blocks of experimental conditions", ylim = c(0, 30))

fit.lm <- lm(y ~ factor(condition) + factor(block), data = blk) anova(fit.aov)

References

Alan S Gerber, Donald P Green (2012). *Field experiments: Design, analysis, and interpretation*. WW Norton.

RB Morton, KC Williams (2010). *Experimental political science and the study of causality: From nature to the lab*. Cambridge University Press.

---

bonett.seier *Bonett-Seier test of Geary's kurtosis*

Description

Performs the Bonett-Seier test of Geary's measure of kurtosis for normally distributed data.

Usage

```
bonett.seier(x, alternative = c("two.sided", "less", "greater"))
```

Arguments

`x`	A numeric vector of data values.
`alternative`	A character string specifying the alternative hypothesis, must be one of '"two.sided"' (default), '"greater"' or '"less"'. You can specify just the initial letter

Details

Under the hypothesis of normality, data should have Geary's kurtosis equal to `sqrt(2/pi)` (0.7979). This test has such null hypothesis and is useful to detect a significant difference of Geary's kurtosis in normally distributed data.

References

Bonett, D.G., Seier, E. (2002) A test of normality with high uniform power. Computational Statistics and Data Analysis, 40, 435-445.

Examples

```
set.seed(1234)
x = rnorm(1000)
geary(x)
bonett.seier(x)
```

`bootstrap` *Method for Bootstrapping*

Description

This method is intended to be provides statistical models that support bootstrapping.

This function is used to estimating standard errors when the distribution is not know.

This method is used to bootstrapping statistical models, typically of class “lm” or “glm”.

Usage

```
bootstrap(x, ...)

## Default S3 method:
bootstrap(x, nboots = 100, FUN, ...)

## S3 method for class 'model'
bootstrap(x, ...)
```

Arguments

`x`	is a vector or a fitted model object whose parameters will be used to produce bootstrapped statistics. Model objects are from the class “glm” or “lm”.
`nboots`	The number of bootstraps.
`FUN`	the name of the statistic to bootstrap, ie., ’mean’, ’var’, ’cov’, etc as a string.
`...`	further arguments passed to or used by other methods.

Value

A list with the “alpha” and “beta” slots set. Note that “alpha” corresponds to ancillary parameters and “beta” corresponds to systematic components of the model.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
x = runif(10, 0, 1)
bootstrap(x,FUN=mean)
```

Bush — *Approval Ratings for President George W. Bush*

Description

Approval ratings for George W. Bush.

Usage

```
data(Bush)
```

Format

A `data.frame` object with 5 variables and 270 observations.

- start.date. Start date of the survey.
- end.date. End date of the survey.
- approve. Percent which approve of the president.
- disapprove. Percent which disapprove of the president.
- undecided. Percent undecided about the president.

butterfly — *The Butterfly Curve*

Description

The butterfly curve is a parametric equation discovered by Temple Fay where two functions in a plane produces butterfly-like curves.

Usage

```
butterfly(n = 100, nb = 500, title = element_blank())
```

Arguments

`n`	An integer for background points.
`nb`	An integer for the butterfly's points.
`title`	A character vector for plot title.

References

Fay, Temple H. (May 1989). The Butterfly Curve. *Amer. Math. Monthly* 96 (5): 442-443. doi:10.2307/2325155.

Examples

```
if (interactive()) {
butterfly(10, 100, title="10 x 100");
butterfly(10, 200, title="10 x 200");
butterfly(10, 500, title="100 x 500");
butterfly(100, 1000, title="100 x 1000");
}
```

calc.CC *Pearson's Contingency Coefficient for Tables*

Description

Compute Pearson's contingency coefficient for an RxC contingency table.

$$C = \sqrt{\frac{T}{N+T}}$$

, where T = the chi-square test statistic and N = the total sample size.

Usage

```
calc.CC(x, y = NULL, ...)

## Default S3 method:
calc.CC(x, y = NULL, ...)
```

Arguments

`x`	A vector or a matrix.
`y`	A vector that is ignored if x is a matrix and required if x is a vector.
`...`	Extra parameters pass to the `table` function.

Details

If we have N observations with two variables where each observation can be classified into one of R mutually exclusive categories for variable one and one of C mutually exclusive categories for variable two, then a cross-tabulation of the data results in a two-way contingency table (also referred to as an RxC contingency table). A common question with regards to a two-way contingency table is whether we have independence. By independence, we mean that the row and column variables are unassociated (i.e., knowing the value of the row variable will not help us predict the value of column variable and likewise knowing the value of the column variable will not help us predict the value of the row variable). One criticism of this statistic is that it does not give a meaningful description of the degree of dependence (or strength of association). That is, it is useful for determining whether there is dependence. However, since the strength of that association also depends on the degrees of freedom as well as the value of the test statistic, it is not easy to interpert the strength of association.

Note

The Cramer contingency coefficient is more commonly used than the Pearson contingency coefficient.

References

Agresti, Alan (1996) *Introduction to categorical data analysis*. NY: John Wiley and Sons. Blaikie, N. 2003. *Analyzing Quantative Data*. London: SAGE.

Examples

```
# some data:
male <- c(33, 76, 6)
female <- c(47, 153, 25)
mat <- cbind( male, female )
rownames(mat) <- c( 'good', 'satisfactory', 'bad')

calc.CC(mat)
```

---

calc.CV *Cramer's V Coefficient for Tables*

Description

Computes the Cramer's V coefficient of association for tables. The Cramer's V is a measure of effect size for a chi-square goodness of fit test.

Usage

```
calc.CV(x, y = NULL, ...)

## Default S3 method:
calc.CV(x, y = NULL, ...)
```

Arguments

`x`	A vector or a matrix.
`y`	A vector that is ignored if x is a matrix and required if x is a vector.
`...`	Extra parameters pass to the `table` function.

Note

Bootstrap confidence intervals for Cramer's V http://support.sas.com/documentation/cdl/en/statugfreq/63124/PDF/default/statugfreq.pdf, p. 1821

References

Agresti, Alan (1996) *Introduction to categorical data analysis*. NY: John Wiley and Sons.

Examples

```
# Consider an experiment with two conditions, each with 100 participants.
# Each participant chooses between one of following three parties.

cond1 <- c(40, 25, 35)
cond2 <- c(25, 35, 45)
mat <- cbind(cond1, cond2)
rownames(mat) <- c( 'party1', 'party2', 'party3')

# To test the null hypothesis that the distribution of preferences
# is identical in the two conditions, we run a chi-square test:
stats::chisq.test(mat) # still significant
```

```
# However, if we want to estimate the effect size, we then use Cramer's V:
calc.CV(mat)

# Agresti (2002), table 3.10, p. 104
# 1991 General Social Survey: The effect size of race on party identification.
gss <- data.frame(
   expand.grid(race=c("black", "white"),
   party=c("dem", "indep", "rep")),
   count=c(103,341,15,105,11,405))

GSS = untable(gss, freq = "count")

calc.CV(GSS$race, GSS$party)
```

`calc.LR` *Likelihood Ratio Test (G test) for Tables*

Description

Computes the likelihood ratio test (G test) for contingency tables. Currently does not do Williams' and Yates' correction.

Usage

```
calc.LR(x, y = NULL, ...)

## Default S3 method:
calc.LR(x, y = NULL, ...)
```

Arguments

x	A vector or a matrix.
y	A vector that is ignored if x is a matrix and required if x is a vector.
...	Extra parameters pass to the `table` function.

References

Agresti, Alan (1996) *Introduction to categorical data analysis*. NY: John Wiley and Sons

Smithson, M.J. (2003) *Confidence Intervals, Quantitative Applications in the Social Sciences Series, No. 140*. Thousand Oaks, CA: Sage. pp. 39-41.

Examples

```
# 2000 General Social Survey-Sex and Party affiliation
# Agresti (1996) p. 38:

gss <- data.frame(
   expand.grid(sex=c("female", "male"),
   party=c("dem", "indep", "rep")),
   count=c(762,484,327,239, 468,477))
```

```
# expand it:
# GSS <- gss[rep(1:nrow(gss), gss[["count"]]),]
GSS = untable(gss, freq = "count")

calc.LR(GSS$party, GSS$sex)
```

---

calc.Phi *The Phi Coefficient for* 2 x 2 *Tables*

Description

Computes the Phi coefficient for 2 x 2 tables.

Usage

```
calc.Phi(x, y = NULL, ...)

## Default S3 method:
calc.Phi(x, y = NULL, ...)
```

Arguments

x	A vector or a matrix.
y	A vector that is ignored if x is a matrix and required if x is a vector.
...	Extra parameters pass to the table function.

Details

Phi is seldom applied for indexing a 2 x 2 table, because the researcher will typically want to contrast the two proportions as an increment or ratio, not with a correlation coefficient. Alternatives to Phi are the Pearson's C; Tschuprow's T, and Cramer's V.

References

Friendly, Michael (2000) *Visualizing Categorical Data*. SAS Institute Inc., p. 63.

Examples

```
# Admission to Berkeley graduate programs:
Berkeley <- data.frame(
expand.grid(GENDER=c("Male", "Female"),
ADMIT=c("Admitted", "Rejected")),
Freq=c(1198,557,1493,1278))

tab = as.table(rbind(c(1198,557), c(1493,1278)))
calc.Phi(tab)
```

calc.TT *Tschuprow's T for Tables*

Description

Computes the Tschuprow's T coefficient of association for tables.

Usage

```
calc.TT(x, y = NULL)

## Default S3 method:
calc.TT(x, y = NULL, ...)
```

Arguments

x	A vector or a matrix.
y	A vector that is ignored if x is a matrix and required if x is a vector.
...	Extra parameters pass to the table function.

Details

Tschuprow's T has the disadvantage of producing an overcorrection. Although kept from being > 1, the correlation coefficient often cannot reach the permissible maximum value of 1. This problem is likely to occur if R is much greater than C (or the other way around) in a large R x C table.

References

Tschuprow, A. A. (1939) *Principles of the Mathematical Theory of Correlation*. Translated by M. Kantorowitsch. W. Hodge & Co.

Examples

```
# some data:
male <- c(33, 76, 6);
female <- c(47, 153, 25);
mat <- cbind( male, female );
rownames(mat) <- c( 'good', 'satisfactory', 'bad');

calc.TT(mat);

# long format
long = untable(mat);

calc.TT(long$Var1, long$Var2)
```

calc.UC *The Uncertainty Coefficient*

Description

The uncertainty coefficient `U(C|R)` measures the proportion of uncertainty (entropy) in the column variable `Y` that is explained by the row variable `X`.

Usage

```
calc.UC(x, y = NULL, direction = c("symmetric", "row", "column"),
  conf.level = NA, p.zero.correction = 1/sum(x)^2, ...)

## Default S3 method:
calc.UC(x, y = NULL, direction = c("symmetric", "row",
  "column"), conf.level = NA, p.zero.correction = 1/sum(x)^2, ...)
```

Arguments

- `x` — A numeric vector, a factor, matrix or data frame.
- `y` — A vector that is ignored if x is a matrix and required if x is a vector.
- `direction` — The direction of the calculation, either `"symmetric"` (default), `"row"`, or `"column"`. `"row"` calculates `uncertainty(R|C)` (column dependent relationship).
- `conf.level` — The confidence level of the interval. If set to NA (which is the default) no confidence interval will be calculated.
- `p.zero.correction` — Slightly nudge zero values so that their logarithm can be calculated.
- `...` — Further arguments are passed to the function `table`, allowing i.e. to set useNA. This refers only to the vector interface.

Details

The uncertainty coefficient is computed as

$$U(C|R) = \frac{H(X) + H(Y) - H(XY)}{H(Y)}$$

and ranges from [0, 1].

Author(s)

Daniel Marcelino, <dmarcelino@live.com>, strongly based on code from Antti Arppe <antti.arppe@helsinki.fi> and Andri Signorell <andri@signorell.net>.

References

Theil, H. (1972), *Statistical Decomposition Analysis*, Amsterdam: North-Holland Publishing Company.

Examples

```
if (interactive()) {
# example from Goodman Kruskal (1954)
m <- as.table(cbind(c(1768,946,115), c(807,1387,438), c(189,746,288), c(47,53,16)));
dimnames(m) <- list(paste("A", 1:3), paste("B", 1:4));
print(m)

calc.UC(m); # default is direction = "symmetric"

calc.UC(m, conf.level=0.95); # direction "symmetric"

calc.UC(m, direction="column");
}
```

---

categories — *Extraction of categorical values as a preprocessing step for making dummy variables*

Description

`categories` stores all the categorical values that are present in the factors and character vectors of a data frame. Numeric and integer vectors are ignored. It is a preprocessing step for the `dummy` function. This function is appropriate for settings in which the user only wants to compute dummies for the categorical values that were present in another data set. This is especially useful in predictive modeling, when the new (test) data has more or other categories than the training data.

Usage

```
categories(x, p = "all")
```

Arguments

`x`	data frame containing factors or character vectors that need to be transformed to dummies. Numerics, dates and integers will be ignored.
`p`	select the top p values in terms of frequency. Either "all" (all categories in all variables), an integer scalar (top p categories in all variables), or a vector of integers (number of top categories per variable in order of appearance.

Value

A list containing the variable names and the categories

Author(s)

Authors: Michel Ballings, and Dirk Van den Poel, Maintainer: `<Michel.Ballings@GMail.com>`

See Also

`dummy`

Examples

```
#create toy data
(traindata <- data.frame(xvar=as.factor(c("a","b","b","c")),
                         yvar=as.factor(c(1,1,2,3)),
                         var3=c("val1","val2","val3","val3"),
                         stringsAsFactors=FALSE))
(newdata <- data.frame(xvar=as.factor(c("a","b","b","c","d","d")),
                       yvar=as.factor(c(1,1,2,3,4,5)),
                       var3=c("val1","val2","val3","val3","val4","val4"),
                       stringsAsFactors=FALSE))

categories(x=traindata,p="all")
categories(x=traindata,p=2)
categories(x=traindata,p=c(2,1,3))
```

`cathedrals` *Cathedrals*

Description

Heights and lengths of Gothic and Romanesque cathedrals. This dataset contains the following columns:

- Type Romanesque or Gothic.
- Height Total height, feet.
- Length Total length, feet.

Usage

```
data(cathedrals)
```

Format

A `data.frame` object with 3 variables and 25 observations.

References

Weisberg, S. (2014). Applied Linear Regression, 4th edition. Hoboken NJ: Wiley.

cgreene76 — *Christensen's and Greene's (1976) Data*

Description

Data set used by Christensen and Greene (1976) on economies of scale in US electric power generation. This dataset contains the following columns:

- firmid Observation id.
- costs Costs in 1970, MM USD.
- output Output, million KwH.
- plabor Price of labor.
- pkap Price of capital.
- pfuel Price of fuel.
- labshr Labor's cost share.
- kapshr Capital's cost share.

Usage

```
data(cgreene76)
```

Format

A `data.frame` object with 8 variables and 99 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Christensen, L. R., and Greene, W. H. (1976) Economies of scale in US electric power generation. *The Journal of Political Economy,* 655–676.

ci — *An S4 Class to Confidence Intervals*

Description

An S4 Class to Confidence Intervals

Calculates the confidence intervals for a vector of data values.

Usage

```
ci(x, level = 0.95, alpha = 1 - level, na.rm = FALSE, ...)

ci(x, level = 0.95, alpha = 1 - level, na.rm = FALSE, ...)
```

Arguments

`x`	A vector of data values.
`level`	The confidence level. Default is `0.95`.
`alpha`	The significance level. Default is `1-level`. If alpha equals 0.05, then your confidence level is 0.95.
`na.rm`	A logical value, default is `FALSE`
`...`	Additional arguements (currently ignored)

Value

`CI lower`	
`Est. Mean`	Mean of data.
`CI upper`	Upper bound of interval.
`Std. Error`	Standard Error of the mean.

Slots

`lower` Lower bound of interval.

`mean` Estimated mean.

`upper` Upper bound of interval.

`stderr` Standard Error of the mean.

Author(s)

Daniel Marcelino, <`dmarcelino@live.com`>.

Examples

```
x <- c(1, 2.3, 2, 3, 4, 8, 12, 43, -1,-4)

ci(x, level=.90)
```

`circularReplicatedSampling`

Replicated Circular-Systematic Sampling

Description

Replicated circular systematic sampling.

Usage

```
circularReplicatedSampling(N = 500, n = 30, g = 6)
```

Arguments

`N`	The population size.
`n`	The sample size.
`g`	Number of independent sub-samples, each containing m = n/g units. Notice that m has to be a multiple of n and g.

Examples

```
circularReplicatedSampling(500, 30, 6)
```

`circularSampling` *Circular Systematic Sampling*

Description

Circular systematic sampling.

Usage

```
circularSampling(N = 500, n = 30)
```

Arguments

`N`	The population size.
`n`	The sample size.

Examples

```
circularSampling(500, 30)
```

`clear` *Clear Memory of All Objects*

Description

This function is a wrapper for the command `rm(list=ls())`.

Usage

```
clear(obj = NULL, keep = TRUE)
```

Arguments

`obj`	The object (as a string) that needs to be removed (or kept)
`keep`	Should `obj` be kept (i.e., everything but `obj` removed)? Or dropped?

Author(s)

Daniel Marcelino

Examples

```
# create objects
a=1; b=2; c=3; d=4; e=5
# remove d
clear("d", keep=FALSE)
ls()
# remove all but a and b
clear(c("a", "b"), keep=TRUE)
ls()
```

`converter` *Unit Converter*

Description

Converts data from a numeric value from one measurement system to another. For instance, distances in miles to kilometers.

Usage

```
converter(x, from, to)
```

Arguments

`x`	A numeric value or vector of data values to be converted.
`from`	A character defining the original unit.
`to`	A character defining the target unit.

Details

`NA` is returned if a conversion cannot be found.

Weight and mass		
Gram	g	metric
Slug	sg	
Pound mass (avoirdupois)	lbm	
U (atomic mass unit)	u	
Ounce mass (avoirdupois)	ozm	
Distance		
Meter	m	metric
Statute mile	mi	
Nautical mile	Nmi	
Inch	in	
Foot	ft	
Yard	yd	
Angstrom	ang	metric
Pica	pica	
Time		
Year	yr	
Day	day	
Hour	hr	
Minute	mn	
Second	sec	
Pressure		
Pascal	Pa (or p)	

Atmosphere	atm (or at)	
mm of Mercury	mmHg	
Force		
Newton	N	metric
Dyne	dyn (or dy)	
Pound force	lbf	
Energy		
Joule	J	metric
Erg	e	
Thermodynamic calorie	c	
IT calorie	cal	metric
Electron volt	eV (or ev)	metric
Horsepower-hour	HPh (or hh)	
Watt-hour	Wh (or wh)	metric
Foot-pound	flb	
BTU	BTU (or btu)	
Power		
Horsepower	HP (or h)	
Watt	W (or w)	metric
Magnetism		
Tesla	T	metric
Gauss	ga	metric
Temperature		
Degree Celsius	C (or cel)	
Degree Fahrenheit	F (or fah)	
Kelvin	K (or kel)	metric
Liquid measure		
Teaspoon	tsp	
Tablespoon	tbs	
Fluid ounce	oz	
Cup	cup	
U.S. pint	pt (or us_pt)	
U.K. pint	uk_pt	
Quart	qt	
Gallon	gal	
Liter	l (or lt)	metric

Examples

```
converter(c(5.6, 6.7), "in", "m")
```

cox.shugart *Cox-Shugart Measure of Proportionality*

Description

Calculate the Cox and Shugart measure of proportionalitybased on a vector of votes and a vector for the electoral outcome. This measure is also referred to as the regression index.

Usage

```
cox.shugart(v, s, ...)

cox.shugart(v, s, ...)
```

Arguments

v	A numeric vector of data values for votes each political party obtained.
s	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
...	Additional arguements (currently ignored)

Value

A single score given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

See Also

inv.cox.shugart, farina, politicalDiversity, grofman, gallagher, lijphart. For more details see the Indices vignette: vignette("Indices", package = "SciencesPo").

Examples

```
if (interactive()) {
# 2012 Queensland state elecion:
pvotes= c(49.65, 26.66, 11.5, 7.53, 3.16, 1.47)
pseats = c(87.64, 7.87, 2.25, 0.00, 2.25, 0.00)

cox.shugart(pvotes, pseats)

# 2012 Quebec provincial election:
pvotes = c(PQ=31.95, Lib=31.20, CAQ=27.05, QS=6.03, Option=1.89, Other=1.88)
pseats = c(PQ=54, Lib=50, CAQ=19, QS=2, Option=0, Other=0)

cox.shugart(pvotes, pseats)
}
```

cronbach — *Cronbach's Alpha for a matrix or data frame*

Description

This function calculates the Cronbach's alpha value of a data frame or matrix.

Usage

```
cronbach(df)
```

Arguments

`df`	`data.frame` or matrix with more than 2 columns.

Value

The Cronbach's alpha value for `df`.

crosstable — *Cross-tabulation*

Description

`crosstable` produces all possible two-way and three-way tabulations of variables.

Usage

```
crosstable(.data, ..., row = TRUE, column = TRUE, deparse.level = 2)

## Default S3 method:
crosstable(.data, ..., row = TRUE, column = TRUE,
  deparse.level = 2)
```

Arguments

`.data`	The data object.
`row`	`TRUE`.
`column`	`TRUE`.
`deparse.level`	Integer controlling the construction of labels in the case of non-matrix-like arguments. If 0, middle 2 rownames, if 1, 3 rownames, if 2, 4 rownames (default).
`...`	The variables for the cross tabulation.

Value

A cross-tabulated object.

See Also

`xtabs`, `Frequency`, `table`, `prop.table`

Examples

```
titanic %>% crosstable( SEX, AGE, SURVIVED)

#' # Agresti (2002), table 3.10, p. 106
# 1992 General Social Survey--Race and Party affiliation
gss <- data.frame(
   expand.grid(Race=c("black", "white"),
   party=c("dem", "indep", "rep")),
   count=c(103,341,15,105,11,405))

df <- gss[rep(1:nrow(gss), gss[["count"]]), ]

crosstable(df, Race, party)

# Tea-Tasting Experiment data
 tea <- data.frame(
   expand.grid(poured=c("Yes", "No"),
   guess=c("Yes", "No")),
   count=c(3,1,1,3))

# nicer way of recreating long tables
data = untable(tea, freq="count")

crosstable(data, poured, guess, row=TRUE, column=TRUE) # fisher=TRUE
```

`css` *Corrected Sum of Squares*

Description

Computes the corrected sum of squares.

Usage

```
css(x, na.rm = TRUE)

## Default S3 method:
css(x, na.rm = TRUE)

## S3 method for class 'data.frame'
css(x, na.rm = TRUE)
```

Arguments

`x`	A numeric vector.
`na.rm`	A logical value indicating whether `NA` values should be stripped before the computation proceeds.

cv — *Pearson's Coefficient of Variation*

Description

Computes the absolute **coefficient of variation cv** as proposed by Karl Pearson. This coefficient is given by the division of the standard deviation by the mean. As the CV reflects a normalized measure of the dispersion of a given probability distribution, values for $cv < 1$ are considered "low-variance", while those with $cv > 1$ "high-variance".

Usage

```
cv(x, na.rm = TRUE, ...)

## Default S3 method:
cv(x, na.rm = TRUE, ...)
```

Arguments

`x`	A numeric vector.
`na.rm`	A logical value, default is `FALSE`
`...`	Additional arguements (currently ignored)

Details

$\frac{sd(x)}{mean(x)} = cv$, which is the inverse of signal-to-noise ratio.

Value

The coefficient of variation.

Author(s)

Daniel Marcelino, `<dmarcelino@live.com>`

See Also

`se`, `skewness`, `kurtosis`, `winsorize`, `outliers`

Examples

```
set.seed(51);
x <- sample(100);
cv(x);
```

database — *Wrapper for dbConnect*

Description

Connects to a SQLite database or creates one if it does not already exist.

Usage

```
database(dbname)
```

Arguments

dbname	A character name or path to database file.

Details

If the '.sqlite' file extension is ommited from the dbname argument it is automatically added.

Value

SQLiteConnection object.

Examples

```
## Not run:
db <- database("mydb")

## End(Not run)
```

dbTempTable — *Creates a temporary table in the database*

Description

This function is useful if most of your work is on a subset of the database

Usage

```
dbTempTable(db, tab_name, query)
```

Arguments

db	a database connection object
tab_name	character name for the teporary table
query	character the query that specifies the temporary table

Details

The table will exist for as long as the database connection is kept open The Select_query argument will take the output from a select_events(sql_only = TRUE) based function

Examples

```
## Not run:
db <- database("Eleicoes")
dbTempTable(db, tab_name = "gerais_2006",
           query = select_events(db, tab = "Candidatos",
           columns = c("nome", "resultado", "partido"),
           where = "data > '1997-01-01'",
           sql_only = TRUE))

## End(Not run)
```

`ddirichlet` *Dirichlet distribution*

Description

Density function and random number generation for the Dirichlet distribution

Usage

```
ddirichlet(x, alpha, log = FALSE, sum = FALSE)
```

Arguments

`x`	a matrix containing observations.
`alpha`	the Dirichlet distribution's parameters. Can be a vector (one set of parameters for all observations) or a matrix (a different set of parameters for each observation), see "Details".
`log`	if `TRUE`, logarithmic densities are returned.
`sum`	if `TRUE`, the (log-)likelihood is returned.

Value

the `ddirichlet` returns a vector of densities (if `sum = FALSE`) or the (log-)likelihood (if `sum = TRUE`) for the given data and alphas.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
mat <- cbind(1:10, 5, 10:1);
mat;
x <- rdirichlet(10, mat);
ddirichlet(x, mat);
```

describe *Statistical description*

Description

Provides description of a vector, matrix, data.frame.

Usage

```
describe(x, ...)
```

Arguments

x	A data frame, matrix, vector, or formula.
...	Additional arguments passed to describe.default.

Examples

```
## Not run:
 describe(turnout)
 desc <- describe(turnout)
 desc$v1   # print description for just v1
 desc[c('v2','v3')]    # print description for two variables.
 desc[sort(names(desc))] # print in alphabetic order by column names.

# Describing part of a data frame:
 with(turnout, describe(v1 ~ v2*v3 + v4) )
 with(turnout, describe(~ v2 + v3) )
 with(turnout, describe(~ v2 + v3, weights=freqs)) # weighted analysis

## End(Not run)
```

destring *Factors to numeric*

Description

Converts factors to numeric like in Stata.

Usage

```
destring(x)
```

Arguments

x	A factor whose levels will be converted.

See Also

safe.chars.

Examples

```
mylevels <- c('Strongly Disagree', 'Disagree', 'Neither', 'Agree', 'Strongly Agree')

myvar <- factor(sample(mylevels[1:5], 10, replace=TRUE))

unclass(myvar) # testing order

destring(myvar)
```

---

detail *Method to Produce Descriptive Statistics Summary*

Description

Method to Produce Descriptive Statistics Summary

This function provides up to 14 statistics for an entire data object: number of cases, mean, standard deviation, variance, standard error, median, mad (median absolute deviation), trimmed and winsorized means, range, minimum, maximum, skewness, and kurtosis. Statistics for a factor variable might be computed based on its 'levels', and is shown accompained whit ans "*".

Usage

```
detail(.data, by = NULL, basic = FALSE, na.rm = TRUE, trim = 0.2,
  type = 2, k = 1)

detail(.data, by = NULL, basic = FALSE, na.rm = TRUE, trim = 0.2,
  type = 2, k = 1)
```

Arguments

.data	a data object (vector or data.frame).
by	a factor variable
basic	indicates if only a short version of the descriptive table should be returned, the default is `basic=TRUE`.
na.rm	a logical value for `na.rm`, default is `na.rm=TRUE`.
trim	is the proportion of the data to be replaced for estimating the average
type	a numeric value (fraction) to be trimmed. The value in trim will be discarded from the top and bottom of data. See in details below
k	a numeric value for observations in the data set to be discarded while computing the winsorized mean. See details below
...	Parameters which are typically ignored

Details

Trimming is not winsorizing. The winsorization process is more complex than simply excluding data. For example, while in a trimmed estimator the extreme values are discarded, in a winsorized estimator, they are rather replaced by certain percentiles.

Value

A data.frame of descriptive statistics

A data frame containing the require computations

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Daniel Marcelino, <dmarcelino@live.com>

References

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S.*. Springer.

Examples

```
#load some data
data(marriage)

# To apply the function
detail(marriage, trim = 0.5, k = 3)
```

---

dHondt — *The D'Hondt Method of Allocating Seats Proportionally*

Description

The function calculate the seats allotment in legislative house, given the total number of seats and the votes for each party based on the Victor D'Hondt's method (1878), which is mathematically equivalent to the method proposed by Thomas Jefferson few years before (1792).

Usage

```
dHondt(parties = NULL, votes = NULL, seats = NULL, ...)

dHondt(parties = NULL, votes = NULL, seats = NULL, ...)
```

Arguments

`parties`	A vector containig parties labels or candidates accordingly to the `votes` vector order.
`votes`	A vector containing the total number of formal votes received by the parties/candidates.
`seats`	An integer for the number of seats to be filled (the district magnitude).
`...`	Additional arguements (currently ignored)

Value

A `data.frame` of length `parties` containing apportioned integers (seats) summing to `seats`.

Note

Adapted from Carlos Bellosta's replies in the R-list.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

References

Lijphart, Arend (1994). *Electoral Systems and Party Systems: A Study of Twenty-Seven Democracies, 1945-1990*. Oxford University Press.

See Also

highestAverages, largestRemainders, hamilton, politicalDiversity.

Examples

```
# Example: 2014 Brazilian election for the lower house in
# the state of Ceara. Coalitions were leading by the
# following parties:

results <- c(DEM=490205, PMDB=1151547, PRB=2449440,
PSB=48274, PSTU=54403, PTC=173151)

dHondt(parties=names(results), votes=results, seats=19)

# The next example is for the state legislative house of Ceara (2014):

votes <- c(187906, 326841, 132531, 981096, 2043217,15061,103679,109830, 213988, 67145, 278267)

parties <- c("PCdoB", "PDT","PEN", "PMDB", "PRB","PSB","PSC", "PSTU", "PTdoB", "PTC", "PTN")

dHondt(parties, votes , seats=42)
```

---

dot.plot *Dot Plot*

Description

Makes a dot plot.

Usage

```
dot.plot(x, pch = 16, bins = 50, spacing = 1, xlab, ...)
```

Arguments

x	The data vector
pch	The plotting "character" or symbol, default is dots.
bins	The bins width.
spacing	A value for vertically spacing between dots.
xlab	The axis label.
...	Other parameters passed on to ‘plot‘.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
with(iris, dot.plot(Sepal.Length,
xlab="Length", col = as.numeric(Species)))
```

---

dotfy *Replace commas by dots*

Description

Replace commas by dots in that order.

Usage

```
dotfy(x)
```

Arguments

x	A vector whose elements contain commas or commas and dots.

Details

This function works for numeric vectors, typically currency variables stored in non-english format.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
x <- c('500,00', '0,001', '25.000', '10,100.10', 'him, you, and I.')

dotfy(x)
```

---

draw.norm *Shades Normal Distribuion*

Description

Produces a plot of a normal density distribution with shaded areas.

Usage

```
draw.norm(below = NULL, above = NULL, pcts = c(0.025, 0.975), mu = 0,
  sigma = 1, numpts = 500, color = "gray", dens = 40,
  justabove = FALSE, justbelow = FALSE, lines = FALSE, between = NULL,
  outside = NULL)
```

Arguments

`below`	sets a lower endpoint.
`above`	sets an upper endpoint.
`pcts`	the
`mu`	the mean.
`sigma`	standard deviations.
`numpts`	the number os points/observations to drawn upon.
`color`	the color of the area.
`dens`	the density of the color.
`justabove`	just plots the upper tail.
`justbelow`	just plots the lower tail.
`lines`	to draw lines.
`between`	plots between specified points.
`outside`	alternative "outside" area.

Value

A plot with a normal distribution density with shaded areas

Examples

```
draw.norm()
draw.norm(below=-1.5)
draw.norm(below=-1.5,justbelow=TRUE)
draw.norm(above=1.5, justabove=TRUE)
draw.norm(below=-1.5,above=1.5)
draw.norm(between=c(-4,0),color="black")
draw.norm(between=c(0,4),color="black")
draw.norm(between=c(-1,+1),color="darkgray")
title("P[-1 < z < 1] = 68%")
draw.norm(between=c(-2,+2),color="darkgray")
title("P[-2 < z < 2] = 95%")
draw.norm(between=c(-3,+3),color="darkgray")
title("P[-3 < z < 3] = 99.7%")
draw.norm(between = c(-1.75, 0, 2, 0.5, -1))  ## Plots between specified points
draw.norm(below=50,justbelow=TRUE,color="black",mu=47.3,sigma=9.3)

## Can plot one and then another on top of it using lines = TRUE
draw.norm(mu=2, sigma=10, outside=c(-3, 12), dens=15)
draw.norm(mu=2, sigma=15, between=c(-3, 12),lines=TRUE, col="blue",dens=15)
## Example: Plotting a Hypothesis Test for the mean
## Truth:      mu.true  = 8
## Hypothesis: mu.ho    = 6
## Generate Data Under Truth
mu.true = 5 ## Alternative Mean
mu.ho   = 6
sig     = 8
N       = 250 ## Sample Size

std.err = sig/sqrt(N)
crits = qnorm(c(0.025,0.975),mean=mu.ho, sd = std.err)
draw.norm(outside = crits, mu = mu.ho, sigma = std.err,dens=15)
draw.norm(between = crits, mu = mu.true, sigma = std.err, lines=TRUE, color="green",dens=15)
```

dummy — *Generate dummy variables*

Description

Provides an alternative to generate dummy variables

Usage

```
dummy(x, data = NULL, drop = TRUE)
```

Arguments

x	a column position to generate dummies
data	the data object as a data.frame
drop	A logical value. If TRUE, unused levels will be omitted

Details

A matrix object

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
df <- data.frame(y = rnorm(25), x = runif(25,0,1), sex = sample(1:2, 25, rep=TRUE))

dummy(df$sex)
```

fade — *Add transparency*

Description

Alpha function to add transparency in graphic objects

Usage

```
fade(color, alpha = 0.5)
```

Arguments

color	Any color or vector of colors
alpha	Level for alpha, default is 0.5

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
# setup data
x <- seq(0, 50, 1)
supply <- x * -2 + 100
demand <- x * 2
# Point size and transparency
plot(supply, demand, pch = 19, cex = 3, col = fade("red", 0.5))
```

---

farina *Farina Index*

Description

Calculates the Farina index also referred to as the cosine proportionality score based on a vector of votes and a vector for the electoral outcome.

Usage

```
farina(v, s, ...)

farina(v, s, ...)
```

Arguments

`v`	A numeric vector of data values for votes each political party obtained.
`s`	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
`...`	Additional arguements (currently ignored)

Value

A single score given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

References

Koppel, M., and A. Diskin. (2009) Measuring disproportionality, volatility and malapportionment: axiomatization and solutions. Social Choice and Welfare 33, no. 2: 281-286.

See Also

`cox.shugart`, `inv.cox.shugart`, `politicalDiversity`, `grofman`, `gallagher`, `lijphart`. For more details see the Indices vignette: `vignette("Indices", package = "SciencesPo")`.

Examples

```
# 2012 Queensland state elecion
pvotes= c(49.65, 26.66, 11.5, 7.53, 3.16, 1.47)
pseats = c(87.64, 7.87, 2.25, 0.00, 2.25, 0.00)

farina(pvotes, pseats)
```

flag — *Add an "id" Variable to a Dataset*

Description

Many functions will not work properly if there are duplicated ID variables in a dataset. This function is a convenience function for `.N` from the "data.table" package to create an "`.id`" variable that when used in conjunction with the existing ID variables, should be unique.

Usage

```
flag(.data, id.vars = NULL)
```

Arguments

`.data`	The input `data.frame` or `data.table`.
`id.vars`	The variables that should be treated as ID variables. Defaults to `NULL`, at which point all variables are used to create the new ID variable.

Value

The input dataset (as a `data.table`) if ID variables are unique, or the input dataset with a new column named "`.id`".

Author(s)

Ananda Mahto

Examples

```
df <- data.frame(A = c("a", "a", "a", "b", "b"),
                 B = c(1, 1, 1, 1, 1), values = 1:5);
df

flag(df, c("A", "B"))

df <- data.frame(A = c("a", "a", "a", "b", "b"),
                   B = c(1, 2, 1, 1, 2), values = 1:5)
df
flag(df, 1:2)
```

freq — *Simple Frequency Table*

Description

Creates a simple frequency data.frame.

Usage

```
freq(x, weighs = NULL, breaks = graphics::hist(x, plot = FALSE)$breaks,
  digits = 3, include.lowest = TRUE, order = c("desc", "asc", "level",
  "name"), useNA = c("no", "ifany", "always"), ...)

## Default S3 method:
freq(x, weighs = NULL, breaks = graphics::hist(x, plot =
  FALSE)$breaks, digits = 3, include.lowest = TRUE, order = c("desc",
  "asc", "level", "name"), useNA = c("no", "ifany", "always"), ...)
```

Arguments

`x`	A vector of values for which the frequency is desired.
`weighs`	A vector of weights.
`breaks`	one of: 1) a vector giving the breakpoints between histogram cells; 2) a function to compute the vector of breakpoints; 3) a single number giving the number of cells for the histogram; 4) a character string naming an algorithm to compute the number of cells (see 'Details'); 5) a function to compute the number of cells.
`digits`	The number of significant digits required.
`include.lowest`	Logical; if `TRUE`, an x[i] equal to the breaks value will be included in the first (or last) category or bin.
`order`	The order method.
`useNA`	Logical; if `TRUE` NA's values are included.
`...`	Additional arguements (currently ignored)

Author(s)

Daniel Marcelino, <`dmarcelino@live.com`>.

See Also

`Frequency`, `crosstable`.

Examples

```
data(Presidents)

freq(Presidents$winner.party)
```

`Frequency` *Frequency Table*

Description

Simulating the FREQ procedure of SPSS.

Usage

```
Frequency(.data, x, verbose = TRUE, ...)

## Default S3 method:
Frequency(.data, x, verbose = TRUE, ...)

Freq(.data, x, verbose = TRUE, ...)
```

Arguments

`.data`	The data.frame.
`x`	A column for which a frequency of values is desired.
`verbose`	A logical value, if `TRUE`, extra statistics are also provided.
`...`	Additional arguements (currently ignored)

See Also

`freq`, `crosstable`.

Examples

```
data(cathedrals)

Frequency(cathedrals, Type)

cathedrals %>% Frequency(Height)
```

`fte_color_pal` *Extended fivethirtyeight.com color palette*

Description

The standard fivethirtyeight.com palette for line plots is blue, red, green. I add an orange ton.

Usage

```
fte_color_pal()
```

See Also

Other colour fte: `scale_color_fte`, `scale_colour_fte`, `scale_fill_fte`

Examples

```
library(scales)
show_col(fte_color_pal()(4))
```

gallagher	*Gallagher Index*

Description

Calculates the Gallagher index of LSq index.

Usage

```
gallagher(v, s, ...)

gallagher(v, s, ...)
```

Arguments

v	A numeric vector of data values for votes each political party obtained.
s	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
...	Additional arguements (currently ignored)

Details

The representation score is calculated as: sqrt(sum((Z-R)^2)/2).

Value

A single score (The Gallagher's Representation Score.) given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

References

Gallagher, M. (1991) Proportionality, disproportionality and electoral systems. Electoral Studies 10(1):33-51.

See Also

cox.shugart, inv.cox.shugart, politicalDiversity, grofman, farina, lijphart. For more details see the Indices vignette: vignette("Indices", package = "SciencesPo")

Examples

```
# 2012 Queensland state elecion
pvotes= c(49.65, 26.66, 11.5, 7.53, 3.16, 1.47)
pseats = c(87.64, 7.87, 2.25, 0.00, 2.25, 0.00)

gallagher(pvotes, pseats)
```

`galton` *Galton's Family Data on Human Stature.*

Description

It is a reproduction of the data set used by Galton in his 1885's paper on correlation between parent's height and their children. However, Galton would only introduce the concept of correlation few years later, in 1888. Galton suggested the use of the regression line and was the first to describe the so-called common phenomenon of regression toward the mean by comparing his experiments on the size of the seeds of successive generations of peas. This dataset contains the following columns:

- parent the parents' average height
- child the child's height

Usage

```
data(galton)
```

Format

A `data.frame` object with 2 variables and 928 observations.

Details

Regression analysis is the statistical method most often used in political science research. The reason is that most scholars are interested in identifying "causal" effects from non-experimental data and that regression is the method for doing this. The term "regresssion" (1889) was first crafted by Sir Francis Galton upon investigating the relationship between body size of fathers and sons. Thereby he "invented" regression analysis by estimating: $S_s = 85.7 + 0.56 S_F$ meaning that the size of the son regresses towards the mean.

References

Francis Galton (1886) Regression Towards Mediocrity in Hereditary Stature. *The Journal of the Anthropological Institute of Great Britain and Ireland,* Vol. **15**, pp. 246–263.

`geary` *Geary's test for normality*

Description

This function computes an estimator of Geary's measure of kurtosis.

Usage

```
geary(x, na.rm = TRUE)
```

Arguments

`x`	the numeric vector.
`na.rm`	A logical for NA values.

Details

Null hypothesis is that the data obeys to normal distribution and that data should have kurtosis equal to 3.

Value

statistic The Geary's test of statistic G.

p.value The significant probability of the null-hypothesis testing.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

Examples

```
set.seed(1234)
x = rnorm(1000)
geary(x)

geary(20:50)

y = c(0.269, 0.357, 0.2, 0.221, 0.275, 0.277, 0.253, 0.127, 0.246)

stats::qqnorm(y)
```

---

geom_foot *Add Footnote to a ggplot Object*

Description

Add footnotes to **ggplot2** objects.

Usage

```
geom_foot(text = NULL, fontsize = 10, color = NULL, rotn = 0,
  just = c("right", "bottom"))
```

Arguments

`text`	any text or empty to use default.
`fontsize`	the font size `text`.
`color`	the color for `text`.
`rotn`	the rotation for the footnote, default is `rotation=90`.
`just`	the justification method.

Details

At this stage, this function only works for a ggplot object.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
# setup data
set.seed(51)
supply <- rnorm(100,mean=15-seq(1,6,by=.05),sd=1)
demand <- rnorm(100,mean=4+seq(1,21,by=.2),sd=.5)
time<-seq(1,100,by=1)
data <- data.frame(time, supply,demand)

# make the plot
library(ggplot2)
ggplot(data,aes(time)) +
geom_line(aes(y=demand),size=1.6, color="#008fd5") +
geom_line(aes(y=supply),size=1.6, color="#ff2700") +
theme_fte() +
annotate("text",x=90,y=12,label="Demand") +
annotate("text",x=80,y=23,label="Supply")
geom_foot("danielmarcelino.github.io", color = "#77ab43", rotn = -90, just ="right" )
```

---

gini *Weighted Gini Index*

Description

Computes the unweighted and weighted Gini index of a distribution.

Usage

```
gini(x, weights = rep(1, length = length(x)), ...)

gini(x, weights = rep(1, length = length(x)), ...)
```

Arguments

x	A data.frame, a matrix-like, or a vector.
weights	A vector containing weights for x.
...	Additional arguements (currently ignored)

Author(s)

Daniel Marcelino, <dmarcelino@live.com> .

See Also

gini.simpson.

Examples

```
# generate a vector (of incomes)
x <- c(778, 815, 857, 888, 925, 930, 965, 990, 1012)
# compute Gini index
gini(x)

gini(c(100,0,0,0))
```

gini.simpson *Gini-Simpson Index*

Description

Computes the Gini/Simpson coefficient. NAs from the data are omitted.

Usage

```
gini.simpson(x, na.rm = TRUE)

gini.simpson(x, na.rm = TRUE)
```

Arguments

x	A data.frame, a matrix-like, or a vector.
na.rm	A logical value to deal with NAs.
...	Additional arguements (currently ignored)

Details

The Gini-Simpson quadratic index is a classic measure of diversity, widely used by social scientists and ecologists. The Gini-Simpson is also known as Gibbs-Martin index in sociology, psychology and management studies, which in turn is also known as the Blau index. The Gini-Simpson index is computed as $1 - \lambda = 1 - \sum_{i=1}^{R} p_i^2 = 1 - 1/^2 D$.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

See Also

politicalDiversity.

Examples

```
# generate a vector (of incomes)
x <- as.table(c(69,50,40,22))
rownames(x) <- c("AB","C","D","E")
gini.simpson(x)
```

griliches76 *Griliches's (1976) Data*

Description

Data set used by Griliches (1976) on wages of very young men. This dataset contains the following columns:

- rns residency in South.
- rns80 a numeric vector.
- mrt marital status = 1 if married.
- mrt80 a numeric vector.
- smsa reside metro area = 1 if urban.
- smsa80 a numeric vector.
- med mother's education, years.
- iq iq score.
- kww score on knowledge in world of work test.
- year Year.
- age a numeric vector.
- age80 a numeric vector.
- s completed years of schooling.
- s80 a numeric vector.
- expr experience, years.
- expr80 a numeric vector.
- tenure tenure, years.
- tenure80 a numeric vector.
- lw log wage.
- lw80 a numeric vector.

Usage

```
data(griliches76)
```

Format

A `data.frame` object with 20 variables and 758 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Griliches, Z. (1976) Wages of very young men. *The Journal of Political Economy,* **84(4),** 69–85.

grofman *Grofman Index*

Description

Calculates the Grofman index of proportionality based on a vector of votes and a vector for the electoral outcome.

Usage

```
grofman(v, s, ...)

grofman(v, s, ...)
```

Arguments

`v`	A numeric vector of data values for votes each political party obtained.
`s`	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
`...`	Additional arguements (currently ignored)

Value

A single score given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

References

Taagepera, R., and B. Grofman. Mapping the indices of seats-votes disproportionality and inter-election volatility. Party Politics 9, no. 6 (2003): 659-77.

See Also

`cox.shugart`, `inv.cox.shugart`, `politicalDiversity`, `farina`, `gallagher`, `lijphart`. For more details see the Indices vignette: `vignette("Indices", package = "SciencesPo")`

Examples

```
# 2012 Quebec provincial election:
pvotes = c(PQ=31.95, Lib=31.20, CAQ=27.05,QS=6.03,Option=1.89, Other=1.88)
pseats = c(PQ=54, Lib=50, CAQ=19, QS=2, Option=0, Other=0)

grofman(pvotes, pseats)
```

hamilton	*The Hamilton Method of Allocating Seats Proportionally*

Description

Computes the Alexander Hamilton's apportionment method (1792), also known as Hare-Niemeyer method or as Vinton's method. The Hamilton method is a largest-remainder method which uses the Hare Quota.

Usage

```
hamilton(parties = NULL, votes = NULL, seats = NULL, ...)

hamilton(parties = NULL, votes = NULL, seats = NULL, ...)
```

Arguments

`parties`	A vector containig parties labels or candidates in the same order of `votes`.
`votes`	A vector with the formal votes received by the parties/candidates.
`seats`	An integer for the number of seats to be returned.
`...`	Additional arguements (currently ignored)

Details

The Hamilton/Vinton Method sets the divisor as the proportion of the total population per house seat. After each state's population is divided by the divisor, the whole number of the quotient is kept and the fraction dropped resulting in surplus house seats. Then, the first surplus seat is assigned to the state with the largest fraction after the original division. The next is assigned to the state with the second-largest fraction and so on.

Value

A `data.frame` of length `parties` containing apportioned integers (seats) summing to `seats`.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

References

Lijphart, Arend (1994). *Electoral Systems and Party Systems: A Study of Twenty-Seven Democracies, 1945-1990*. Oxford University Press.

See Also

`dHondt`, `highestAverages`, `largestRemainders`, `politicalDiversity`.

Examples

```
votes <- sample(1:10000, 5)
parties <- sample(LETTERS, 5)
hamilton(parties, votes, seats = 4)
```

has.domain *Find WWW Domains*

Description

Hand function to return the www domain.

Usage

```
has.domain(x)
```

Arguments

x	A vector from which the domain information is desired.

Examples

```
x1 <- "http://stackoverflow.com/questions/19020749/function-to-extract-domain-name-from-url-in-r"
x2 <- "http://www.talkstats.com/"
x3 <- "www.google.com"

has.domain(x3)

sapply(list(x1, x2, x3), has.domain)
```

herfindahl *Herfindahl Index of Concentration*

Description

Calculates the Herfindahl Index of concentration.

Usage

```
herfindahl(x, n = rep(1, length(x)), parameter = 1, na.rm = FALSE, ...)

herfindahl(x, n = rep(1, length(x)), parameter = 1, na.rm = FALSE, ...)
```

Arguments

x	A vector of data values of non-negative elements.
n	A vector of frequencies of the same length as x.
parameter	A parameter of the concentration measure (if set to NULL the default parameter of the respective measure is used).
na.rm	A logical. Should missing values be removed? The Default is set to na.rm=FALSE.
...	Additional arguements (currently ignored)

Details

This index is also known as the *Simpson Index* in ecology, the *Herfindahl-Hirschman Index (HHI)* in economics, and as the *Effective Number of Parties (ENP)* in political science.

References

Cowell, F. A. (2000) Measurement of Inequality in Atkinson, A. B. / Bourguignon, F. (Eds): *Handbook of Income Distribution*. Amsterdam.

Cowell, F. A. (1995) *Measuring Inequality* Harvester Wheatshef: Prentice Hall.

See Also

atkinson, rosenbluth, politicalDiversity, gini. For more details see the Indices vignette: `vignette("Indices", package = "SciencesPo")`.

Examples

```
# generate a vector (of incomes)
x <- c(778, 815, 857, 888, 925, 930, 965, 990, 1012)

# compute the Herfindahl coefficient with parameter=1
herfindahl(x, parameter=1)
```

---

highestAverages — *Highest Averages Methods of Allocating Seats Proportionally*

Description

Computes the highest averages method for a variety of formulas of allocating seats proportionally.

Usage

```
highestAverages(parties = NULL, votes = NULL, seats = NULL,
  method = c("dh", "sl", "msl", "danish", "hsl", "hh", "imperiali", "wb",
  "jef", "ad", "hb"), threshold = 0, ...)

## Default S3 method:
highestAverages(parties = NULL, votes = NULL,
  seats = NULL, method = c("dh", "sl", "msl", "danish", "hsl", "hh",
  "imperiali", "wb", "jef", "ad", "hb"), threshold = 0, ...)
```

Arguments

`parties`	A character vector for parties labels or candidates in the same order as `votes`. If `NULL`, alphabet will be assigned.
`votes`	A numeric vector for the number of formal votes received by each party or candidate.
`seats`	The number of seats to be filled (scalar or vector).
`method`	A character name for the method to be used. See details.
`threshold`	A numeric value between (0~1). Default is set to 0.
`...`	Additional arguements (currently ignored)

Details

The following methods are available:

- "dh"d'Hondt method
- "sl"Sainte-Lague method
- "msl"Modified Sainte-Lague method
- "danish"Danish modified Sainte-Lague method
- "hsl"Hungarian modified Sainte-Lague method
- "imperiali"The Italian Imperiali (not to be confused with the Imperiali quota which is a Largest remainder method)
- "hh"Huntington-Hill method
- "wb"Webster's method
- "jef"Jefferson's method
- "ad"Adams's method
- "hb"Hagenbach-Bischoff method

Value

A `data.frame` of length `parties` containing apportioned integers (seats) summing to `seats`.

Author(s)

Daniel Marcelino, <`dmarcelino@live.com`>.

References

Gallagher, Michael (1992). "Comparing Proportional Representation Electoral Systems: Quotas, Thresholds, Paradoxes and Majorities". *British Journal of Political Science*, 22, 4, 469-496.

Lijphart, Arend (1994). *Electoral Systems and Party Systems: A Study of Twenty-Seven Democracies, 1945-1990*. Oxford University Press.

See Also

`largestRemainders`, `dHondt`, `hamilton`, `politicalDiversity`. For more details see the *Indices* vignette: `vignette('Indices', package = 'SciencesPo')`.

Examples

```
# Results for the state legislative house of Ceara (2014):
votes <- c(187906, 326841, 132531, 981096, 2043217, 15061, 103679,109830, 213988, 67145, 278267)

parties <- c("PCdoB", "PDT", "PEN", "PMDB", "PRB", "PSB", "PSC", "PSTU", "PTdoB", "PTC", "PTN")

highestAverages(parties, votes, seats = 42, method = "dh")

# Let's create a data.frame with typical election results
# with the following parties and votes to return 10 seats:

my_election <- data.frame(
party=c("Yellow", "White", "Red", "Green", "Blue", "Pink"),
votes=c(47000, 16000,15900,12000,6000,3100))
```

```
highestAverages(my_election$party,
my_election$votes,
seats = 10,
method="dh")

# How this compares to the Sainte-Lague Method

(dat= highestAverages(my_election$party,
my_election$votes,
seats = 10,
method="sl"))

# Plot it
bar.plot(data=dat, "Party", "Seats") +
theme_fte()
```

`insert.row` *Add new row to dataframe*

Description

Facilitates insertion of a new row in a data.frame.

Usage

```
insert.row(.data, newrow, where = 1)
```

Arguments

`.data`	The existing data.frame
`newrow`	The new row to be appended.
`where`	An integer for the position to add the row, default is at the top.

Examples

```
existingDF <- as.data.frame(matrix(seq(20),nrow=5,ncol=4))
existingDF
r <- 3
newrow <- seq(4)

insert.row(existingDF, newrow, r)
```

`inv.cox.shugart` *Inverse Cox-Shugart Measure of Proportionality*

Description

Calculate the inverse Cox and Shugart measure of proportionality based on votes and seats, the electoral outcome.

Usage

```
inv.cox.shugart(v, s, ...)

inv.cox.shugart(v, s, ...)
```

Arguments

`v`	A numeric vector of data values for votes each political party obtained.
`s`	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
`...`	Additional arguements (currently ignored)

Value

A single score given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

See Also

`cox.shugart`, `farina`, `politicalDiversity`, `grofman`, `gallagher`, `lijphart`. For more details see the Indices vignette: `vignette("Indices", package = "SciencesPo")`.

Examples

```
# 2012 Queensland state elecion:
pvotes= c(49.65, 26.66, 11.5, 7.53, 3.16, 1.47)
pseats = c(87.64, 7.87, 2.25, 0.00, 2.25, 0.00)

inv.cox.shugart(pvotes, pseats)

# 2012 Quebec provincial election:
pvotes = c(PQ=31.95, Lib=31.20, CAQ=27.05, QS=6.03, Option=1.89, Other=1.88)
pseats = c(PQ=54, Lib=50, CAQ=19, QS=2, Option=0, Other=0)

inv.cox.shugart(pvotes, pseats)
```

`invnormal` *Inverse Cumulative Standard Normal Distribution*

Description

Computes the inverse cumulative distribution of `x` associated with an *area* under the normal distribution curve given by μ and standard deviation σ.

Usage

```
invnormal(area, mu = 0, sigma = 1)
```

Arguments

`area`	the area or a vector of probabilities.
`mu`	the mean μ.
`sigma`	the standard deviation of the distribution σ.

Author(s)

Daniel Marcelino, `<dmarcelino@live.com>`

See Also

`draw.norm`, `normalpdf`, `normalcdf`

Examples

```
invnormal(area=0.35,mu=0,sigma=1)
```

`jackknife` *Resamples Data Using the Jackknife Method*

Description

This function is used for estimating standard errors when the distribution is not know.

Usage

```
jackknife(x, p)
```

Arguments

`x`	A vector
`p`	An function name for estimation of parameter as a string

Value

est orignial estimation of parameter

jkest jackknife estimation of parameter

jkvar jackknife estimation of variance

jkbias jackknife estimate of biasness of parameter

jkbiascorr bias corrected parameter estimate

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
x = runif(10, 0, 1)
mean(x)
jackknife(x,'mean')
```

`james.stein` *James-Stein shrunken estimates*

Description

Computes James-Stein shrunken estimates of cell means given a response variable (which may be binary) and a grouping indicator.

Usage

```
james.stein(y, k)
```

Arguments

`y`	The response variable.
`k`	The grouping factor.

References

Efron, Bradley and Morris, Carl (1977) “Stein’s Paradox in Statistics.” *Scientific American* Vol. 236 (5): 119-127.

James, Willard and Stein, Charles (1961) “Estimation with Quadratic Loss.” *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1: 361-379.

`jarque.bera` *Jarque-Bera test for normality*

Description

This function performs the Jarque-Bera test on the given data sample to determine if the data are sample drawn from a normal population.

Usage

```
jarque.bera(x)
```

Arguments

`x`	A numeric vector of data.

Details

The Jarque-Bera statistic is chi-square distributed with two degrees of freedom. Under the hypothesis of normality, data should be symmetrical (i.e. skewness should be equal to zero) and have skewness chose to three.

References

Jarque, C. M., Bera, A. K. (1980) Efficient test for normality, homoscedasticity and serial independence of residuals, Economic Letters, Vol. 6 Issue 3, 255-259.

Examples

```
set.seed(1234)

x <- rnorm(1000)

jarque.bera(x)
```

`jensen.shannon` *Jensen-Shannon Distance*

Description

The Jensen-Shannon divergence or distance matrix stores the $n*(n-1)/2$ pairwise distances/similarities between observations in an nxp matrix where n correspond to the independent observational units and p represent the covariates measured on each individual.

Usage

```
jensen.shannon(mat)
```

Arguments

`mat`	An n x p matrix.

Examples

```
# create a matrix
n  = 10
m = matrix(runif(n*10), ncol = 10)
m = m/rowSums(m)

jensen.shannon(m)
```

---

johnson.neyman *Johnson-Neyman Regression*

Description

Probing Regression Interactions

Usage

```
johnson.neyman(y, x, z)
```

Arguments

y	the dependent variable.
x	the independent variable.
z	the moderator variable.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

---

kurtosis *Compute the Kurtosis*

Description

Return the kurtosis test for object x. For vectors, kurtosis(x) is the kurtosis of the elements in the vector x. For matrices kurtosis(x) returns the sample kurtosis for each column of x. For N-dimensional arrays, kurtosis operates along the first nonsingleton dimension of x.Returns the kurtosis test for object x. For vectors, kurtosis(x) is the kurtosis of the elements in the vector x. For matrices kurtosis(x) returns the sample kurtosis for each column of x. For N-dimensional arrays, kurtosis operates along the first nonsingleton dimension of x.

Usage

```
kurtosis(x, na.rm = FALSE, type = 2)
```

Arguments

x	a numeric vector
na.rm	a logical value for na.rm, default is na.rm=FALSE.
type	an integer between 1 and 3 selecting one of the algorithms for computing kurtosis detailed below

Details

In a similar way of skewness, kurtosis measures the peakedness of a data distribution. A distribution with zero kurtosis has a shape as the normal curve. Such type of kurtosis is called mesokurtic, or mesokurtotic. A positive kurtosis has a curve more peaked about the mean and the its shape is narrower than the normal curve. Such type is called leptokurtic, or leptokurtotic. Finally, a distribution with negative kurtosis has a curve less peaked about the mean and the its shape is flatter than the normal curve. Such type is called platykurtic, or platykurtotic. To be consistent with classical use of kurtosis in political science analyses, the default **type** is the same equation used in SPSS and SAS, which is the bias-corrected formula: **Type 2:** G_2 = ((n + 1) g_2+6) * (n-1)/(n-2)(n-3). When you set type to 1, the following equation applies: **Type 1:** g_2 = m_4/m_2^2-3. When you set type to 3, the following equation applies: **Type 3:** b_2 = m_4/s^4-3 = (g_2+3)(1-1/n)^2-3. You must have at least 4 observations in your vector to apply this function.

Value

An object of the same type as `x`.

Note

Skewness and **Kurtosis** are functions to measure the third and fourth **central moment** of a data distribution.

Author(s)

Daniel Marcelino, `<dmarcelino@live.com>`

References

Balanda, K. P. and H. L. MacGillivray. (1988) Kurtosis: A Critical Review. *The American Statistician,* **42(2), pp. 111–119.**

Examples

```
w<-sample(4,10, TRUE)
x <- sample(10, 1000, replace=TRUE, prob=w)

kurtosis(x, type=2)

kurtosis(x, type=3)
```

`largestRemainders` *Largest Remainders Methods of Allocating Seats Proportionally*

Description

Computes the largest remainders method for a variety of formulas of allocating seats proportionally.

Usage

```
largestRemainders(parties = NULL, votes = NULL, seats = NULL,
  method = c("dh", "sl", "msl", "danish", "hsl", "hh", "imperiali", "wb",
  "jef", "ad", "hb"), threshold = 0, ...)

## Default S3 method:
largestRemainders(parties = NULL, votes = NULL,
  seats = NULL, method = c("dh", "sl", "msl", "danish", "hsl", "hh",
  "imperiali", "wb", "jef", "ad", "hb"), threshold = 0, ...)
```

Arguments

`parties`	A character vector for parties labels or candidates in the order as `votes`. If `NULL`, a random combination of letters will be assigned.
`votes`	A numeric vector for the number of formal votes received by each party or candidate.
`seats`	The number of seats to be filled (scalar or vector).
`method`	A character name for the method to be used. See details.
`threshold`	A numeric value between (0~1). Default is set to 0.
`...`	Additional arguements (currently ignored)

Details

The following methods are available:

- "dh"d'Hondt method
- "sl"Sainte-Lague method

Value

A `data.frame` of length `parties` containing apportioned integers (seats) summing to `seats`.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

References

Gallagher, Michael (1992). "Comparing Proportional Representation Electoral Systems: Quotas, Thresholds, Paradoxes and Majorities". *British Journal of Political Science*, 22, 4, 469-496.

Lijphart, Arend (1994). *Electoral Systems and Party Systems: A Study of Twenty-Seven Democracies, 1945-1990*. Oxford University Press.

See Also

`highestAverages`, `dHondt`, `hamilton`, `politicalDiversity`. For more details see the *Indices* vignette: `vignette('Indices', package = 'SciencesPo')`.

Examples

```
# Let's create a data.frame with typical election results
# with the following parties and votes to return 10 seats:

my_election <- data.frame(
party=c("Yellow", "White", "Red", "Green", "Blue", "Pink"),
votes=c(47000, 16000,15900,12000,6000,3100))

largestRemainders(my_election$party,
my_election$votes, seats = 10,  method="droop")
```

`levy.flight` *Simulates a Levy Walk*

Description

This function simulates a Levy walk

Usage

```
levy.flight(n = 500, alpha = 3, min.lenght = 1, max.lenght = 5,
  plot = TRUE)
```

Arguments

`n`	The lenght of walk.
`alpha`	The exponent of the Levy distribution.
`min.lenght`	The minimum length of a step.
`max.lenght`	The maximum length of a step.
`plot`	A logical, TRUE will make a plot.

Author(s)

Daniel Marcelino, <dmarcelino@live.com> #'

Examples

```
levy.flight(n=100, alpha=2)
```

lijphart *Lijphart Index of Proportionality*

Description

Calculates the Lijphart index of proportionality based on a vector of votes and a vector for the electoral outcome.

Usage

```
lijphart(v, s, ...)

lijphart(v, s, ...)
```

Arguments

v	A numeric vector of data values for votes each political party obtained.
s	A numeric vector of data values for seats each political party obtained, the election outcome as seats.
...	Additional arguements (currently ignored)

Value

A single score given the votes each party received and seats obtained.

Author(s)

Daniel Marcelino <dmarcelino@live.com>

See Also

cox.shugart, inv.cox.shugart, politicalDiversity, grofman, gallagher, farina. For more details see the Indices vignette: vignette("Indices", package = "SciencesPo")

Examples

```
# 2012 Quebec provincial election:
pvotes = c(PQ=31.95, Lib=31.20, CAQ=27.05,QS=6.03,Option=1.89, Other=1.88)
pseats = c(PQ=54, Lib=50, CAQ=19, QS=2, Option=0, Other=0)

lijphart(pvotes, pseats)
```

`lilliefors` *Lilliefors (Kolmogorov-Smirnov) test for normality*

Description

Performs the Lilliefors (Kolmogorov-Smirnov) test for the composite hypothesis of normality. The Lilliefors (Kolomorov-Smirnov) test is the most famous EDF omnibus test for normality; compared to the Anderson-Darling test and the Cramer-von Mises test it is known to perform worse.

Usage

```
lilliefors(x)
```

Arguments

`x`	A numeric vector of data values, the number of observations must be greater than 4. Missing values are allowed.

References

Thode Jr., H.C. (2002): Testing for Normality. Marcel Dekker, New York.

Examples

```
set.seed(1234)
x = rnorm(1000)
lilliefors(x)
```

`linearReplicatedSampling` *Replicated-Systematic Random Sampling*

Description

Replicated-Systematic Random Sampling.

Usage

```
linearReplicatedSampling(N = 500, n = 30, g = 6)
```

Arguments

`N`	The population size.
`n`	The sample size.
`g`	Number of independent sub-samples, each containing m = n/g units. Notice that m has to be a multiple of n and g.

Examples

```
linearReplicatedSampling(500, 30, 6)
```

`linearSampling` *Linear Systematic Sampling*

Description

Linear systematic sampling.

Usage

```
linearSampling(N = 500, n = 30)
```

Arguments

`N`	The population size.
`n`	The sample size.

Examples

```
linearSampling(500, 30)
```

`lm2eqn` *Linear model to equation style*

Description

Produces a text equation style to be added in plots.

Usage

```
lm2eqn(.data, x, y, spaced = TRUE)
```

Arguments

`.data`	The data.frame object.
`x`	The independent variable(s).
`y`	The dependent variable.
`spaced`	A logical value indicating if spaces should be added; default is TRUE.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
lm2eqn("mtcars","wt","mpg")


data(Presidents)

Presidents <- transform(Presidents, ratio = winner.height/opponent.height)
Presidents <- transform(Presidents, selected = ifelse(winner %in% c("Barack Obama"),1,0))

# subsetting election > 1824
Presidents = subset(Presidents, election > 1824 & !is.na(ratio))

selected=Presidents[Presidents$selected==1,]
myeqn=lm2eqn("Presidents","ratio","winner.vote")

ggplot(Presidents, aes(x=ratio,y=winner.vote,colour=selected)) +
 geom_text(data=selected,aes(label=winner),hjust=-0.1) +
 geom_smooth(method=lm, colour="red", fill="gold") +
 geom_point(size=5, alpha=.7) +
annotate(geom='text',x=1.1,y=64,size=7,label=myeqn,family='Times',fontface='italic') +
xlim(.9,1.2) + ylim(38, 65) +
 scale_colour_gradientn(guide="none" , colours=c("black","red")) +
xlab("Presidential Height Ratio") +
 ylab("Relative Support for President")
```

---

lorenz *The Lorenz Curve*

Description

Computes the (empirical) ordinary and generalized Lorenz curve of a vector.

Usage

```
lorenz(x, n = rep(1, length(x)), plot = FALSE, ...)

lorenz(x, n = rep(1, length(x)), plot = FALSE, ...)
```

Arguments

x	A vector of non-negative values.
n	A vector of frequencies of the same length as x.
plot	A logical. If TRUE the Lorenz curve will be plotted.
...	Additional arguements (currently ignored)

Details

The Gini coefficient ranges from a minimum value of zero, when all individuals are equal, to a theoretical maximum of one in an infinite population in which every individual except one has a size of zero. It has been shown that the sample Gini coefficients originally defined need to be multiplied by n/(n-1) in order to become unbiased estimators for the population coefficients.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

See Also

`gini`, `gini.simpson`.

Examples

```
# generate a vector (of incomes)
x <- c(778, 815, 857, 888, 925, 930, 965, 990, 1012)
# compute Lorenz values
lorenz(x)
# generate some weights:
wgt <- runif(n=length(x))
# compute the lorenz with especific weights
lorenz(x, wgt)
```

`ltaylor96` *Lothian's and Taylor's (1996) Data Set*

Description

Data used by Lothian and Taylor (1996) on the real exchange rate behaviour. This dataset contains the following columns:

- year Year
- spot dollar/sterling exchange rate.
- USwpi U.S. wholesale price index, 1914==100.
- UKwpi U.K. wholesale price index, 1914==100.

Usage

```
data(ltaylor96)
```

Format

A `data.frame` object with 4 variables and 200 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Lothian, J. R., and Taylor, M. P. (1996) Real exchange rate behavior: the recent float from the perspective of the past two centuries. *Journal of Political Economy,* 488–509.

marriage *Same Sex Marriage Public Opinion Data*

Description

Data set fielded by the PEW Research Center on same sex marriage support in US. It covers public opinion on the issue starting from 1996 up to date. This dataset contains the following columns:

- Date The year of the measurement
- Oppose Percent opposing same-sex marriage
- Favor Percent favoring same-sex marriage
- DK Percent of Don't Know

Usage

```
data(marriage)
```

Format

A `data.frame` object with 4 variables and 18 observations.

References

PEW Research Center. *Support for same-sex marriage*. http://www.pewresearch.org

meanFromRange *Estimates Mean and Standard Deviation from the Median and the Range*

Description

When conductig a meta-analysis study, it is not always possible to recover from reports the mean and standard deviation values, but rather the medians and range of values. This function provides an approach to convert the median/range into a mean and a variance.

Usage

```
meanFromRange(low, med, high, n)
```

Arguments

`low`	The min of the data.
`med`	The median of the data.
`high`	The max of the data
`n`	The size of the sample.

References

Hozo1, Stela P.; et al (2005) Estimating the mean and variance from the median, range, and the size of a sample. *BMC Medical Research Methodology*, 5:13.

Examples

```
meanFromRange(5,8,12,10)
```

`mishkin92` *Mishkin's (1992) Data*

Description

Data from the Frederic S. Mishkin (1992) paper "Is the Fisher Effect for real?". This dataset contains the following columns:

- year Year
- mon a numeric vector
- inf1mo a numeric vector
- inf3mo a numeric vector
- tbill1mo a numeric vector
- tbill3mo a numeric vector
- cpiu a numeric vector
- quote a numeric vector

Usage

```
data(mishkin92)
```

Format

A `data.frame` object with 8 variables and 491 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Mishkin, F. S. (1992) Is the Fisher effect for real?: A reexamination of the relationship between inflation and interest rates. *Journal of Monetary Economics,* **30(2),** 195–215.

`Mode` *Calculate the Mode*

Description

Estimates the mode for a vector

Usage

```
Mode(x, na.rm = FALSE)

Mode(x, na.rm = FALSE)

Mode.data.frame(x, na.rm = TRUE)
```

Arguments

`x`	An R object.
`na.rm`	A logical value indicating whether `NA` should be stripped before the computation proceeds. Default is `FALSE`

Note

This function replaces the `base` function of the same name, while `SciencesPo::mode` calculates the "mode", `base::mode` prints the "class" of an object.

Author(s)

Daniel Marcelino, <`dmarcelino@live.com`>

Examples

```
myvar <-c(1,1,2,2,3,3,4,4,5, NA)
Mode(myvar)

Mode(myvar, FALSE)
```

`nerlove63` *Marc Nerlove's (1963) data*

Description

Data used by Marc Nerlove (1963) on returns of electricity supply. This dataset contains the following columns:

- totcost costs in 1970, MM USD.
- output output, billion KwH.
- plabor price of labor.
- pfuel price of fuel.
- pkap price of capital.

#'

Usage

```
data(nerlove63)
```

Format

A `data.frame` object with 5 variables and 145 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Nerlove, M. (1963) Returns to Scale in Electricity Supply. In *Measurement in Economics-Studies in Mathematical Economics and Econometrics in Memory of Yehuda Grunfeld,* edited by Carl F. Christ. Stanford: Stanford.

---

normalcdf — *Normal Cumulative Distribution*

Description

Calculates the normal distribution probability using *lower bound* e *upper bound* by the mean μ and standard deviation.

Usage

```
normalcdf(lower, upper, mu = 0, sigma = 1)
```

Arguments

`lower`	is the inferior extreme value.
`upper`	is the superior extreme value.
`mu`	is the mean μ, its default value is $\mu = 0$
`sigma`	is the standard deviation σ, its default value is $\sigma = 1$

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

See Also

`draw.norm`, `normalpdf`, `invnormal`.

Examples

```
normalcdf(lower=-1.96,upper=1.96,mu=0,sigma=1)
```

`normalize` *Unity-based normalization*

Description

Normalizes as feature scaling `min - max`, or unity-based normalization typically used to bring the values into the range [0,1].

Usage

```
normalize(x, method = "range")

## S4 method for signature 'ANY'
normalize(x, method = "range")
```

Arguments

`x`	is a vector to be normalized.
`method`	A string for the method used for normalization. Default is `method = "range"`, which brings the values into the range [0,1]. See details for other implemented methods.

Details

This approach may also be generalized to restrict the range of values to any arbitrary values `a` and `b`, using:

$$X' = a + \frac{(x - x_{min})(b - a)}{(x_{max} - x_{min})}$$

.

Value

Normalized values in an object of the same class as `x`.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

See Also

`svTransform`, `scale`.

Examples

```
x <- sample(10)
normalize(x)

# equivalently to
(x-min(x))/(max(x)-min(x))
```

normalpdf	*Normal probability density function*

Description

Computes the pdf at each of the values in x using the normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$.

Usage

```
normalpdf(x, mu = 0, sigma = 1)
```

Arguments

x	a vector of quantiles.
mu	is the mean μ, its default value is $\mu = 0$
sigma	is the standard deviation σ, its default value is $\sigma = 1$

Note

The pdf function is given by:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$$

for $\sigma > 0$

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

See Also

draw.norm, normalcdf, invnormal. #'

Examples

```
normalpdf(x=1.2,mu=0,sigma=1)
```

`outliers` *Detect Outliers*

Description

Perform an exploaratory test to detect *outliers*. The quantity for *min* reveals the minimum deviation from the mean, the integer in *closest* highlights the position of the element. The quantity for *max* is the maximum deviation from the mean, and the `farthest` integer is the position of such higher quantity.

Usage

```
outliers(x, index = NULL)
```

Arguments

`x`	A numeric object
`index`	A numeric value to be considered in the computations

Value

Returns the minimum and maximum values, respectively preceded by their positions in the `vector`, `matrix` or `data.frame`.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Dixon, W.J. (1950) Analysis of extreme values. *Ann. Math. Stat.* **21(4),** 488–506.

See Also

winsorize for diminishing the impact of outliers.

Examples

```
outliers(x <- rnorm(20))

#data frame:
age <- sample(1:100, 1000, rep=TRUE);
outliers(age)
```

paired — *Paired Data*

Description

Artificial data for a paired experiment. This dataset contains the following columns:

- patient the patient id.
- before_X before treatment.
- after_Y after treatment.

Usage

```
data(paired)
```

Format

A `data.frame` object with 3 variables and 9 observations.

parties_color_pal — *Color Palettes for Political Organizations (discrete)*

Description

Color palettes for political organizations.

Usage

```
parties_color_pal(palette = "BRA")
```

Arguments

`palette`	Palette name.

See Also

Other colour parties: `scale_color_parties`, `scale_colour_parties`, `scale_fill_parties`

Examples

```
library(scales)
show_col(parties_color_pal()(10))
```

`pause` *Pause*

Description

A replication of MatLab pause function.

Usage

```
pause(x = 0)
```

Arguments

x	is optional. If x>0 a call is made to Sys.sleep. Else, execution pauses until a key is entered.

`permutate` *Create k random permutations of a vector*

Description

Creates a k random permutation of a vector.

Usage

```
permutate(input, k)
```

Arguments

input	A vector to be permutated.
k	number of permutations to be conducted.

Details

should be used only for length(input)! » k

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

Examples

```
# row wise permutations
permutate(input=1:5, k=5)
```

pie.plot — *Creates a pie chart using ggplot2.*

Description

Use pie charts with care. See http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=00018S on Edward Tufte's website for good arguments against the use of pie charts. For a contrary point-of-view, see Spence's article, No Humble Pie: The Origins and Usage of a Statistical Chart (http://www.psych.utoronto.ca/us

Usage

```
pie.plot(.data, var, label = var)
```

Arguments

`.data`	the data frame.
`var`	the name of the column to generate the pie chart for.
`label`	The label for the legend.

Examples

```
if (interactive()) {
x = sample(10, 100, rep = TRUE)
z = sample(letters[1:3],100, rep=TRUE)
dat = data.frame(x,z)
pie.plot(dat, 'x', 'z')
}
```

plotTitleSubtitle — *Add Title and Subtitle to a ggplot Object*

Description

A production function to make it easy to add title and subtitle to **ggplot2** objects.

Usage

```
plotTitleSubtitle(title, subtitle = "")
```

Arguments

`title`	A character string as title.
`subtitle`	A character string as subtitle.

Examples

```
# setup data
set.seed(51)
supply <- rnorm(100,mean=15-seq(1,6,by=.05),sd=1)
demand <- rnorm(100,mean=4+seq(1,21,by=.2),sd=.5)
time<-seq(1,100,by=1)
data <- data.frame(time, supply,demand)

# make the plot
library(ggplot2)
ggplot(data,aes(time)) +
geom_line(aes(y=demand),size=1.6) +
geom_line(aes(y=supply),size=1.6) +
annotate("text",x=90,y=12,label="Demand",colour="red") +
annotate("text",x=80,y=23,label="Supply",colour="blue") +
plotTitleSubtitle("My Title", "My Subtitle")
```

---

politicalDiversity *Political Diversity Indices*

Description

Analyzes political diversity in an electoral unity or across unities. It provides methods for estimating the effective number of parties and other fragmentation/concetration measures. The intuition of these coefficients is to counting parties while weighting them by their relative political–or electoral strength.

Usage

```
politicalDiversity(x, index = "laakso/taagepera", margin = 1,
  base = exp(1))

## S4 method for signature 'ANY'
politicalDiversity(x, index = "laakso/taagepera",
  margin = 1, base = exp(1))
```

Arguments

x	A data.frame, a matrix-like, or a vector containing values for the number of votes or seats each party received.
index	The type of index desired, one of "laakso/taagepera", "golosov", "herfindahl", "gini", "shannon", "simpson", "invsimpson".
margin	The margin for which the index is computed.
base	The logarithm base used in some indices, such as the "shannon" index.

Details

Very often, political analysts say things like ‘two-party system’ and ‘multi-party system’ to refer to a particular kind of political party system. However, these terms alone does not tell exactly how fragmented–or concentrated a party system actually is. For instance, after the 2010 general election, 22 parties obtained representation in the Lower Chamber in Brazil. Nonetheless, among these 22 parties, nine parties together returned only 28 MPs. Thus, an index to assess the weigh or the **Effective Number of Parties** is important and helps to go beyond the simple count of parties in a legislative branch.

A widely accepted algorithm was proposed by M. Laakso and R. Taagepera:

$$N = \frac{1}{\sum p_i^2}$$

, where **N** denotes the effective number of parties and **p_i** denotes the it^h party's fraction of the seats.

In fact, this formula may be used to compute the vote share for each party. This formula is the reciprocal of a well-known concentration index (**the Herfindahl-Hirschman index**) used in economics to study the degree to which ownership of firms in an industry is concentrated. Laakso and Taagepera correctly saw that the effective number of parties is simply an instance of the inverse measurement problem to that one. This index makes rough but fairly reliable international comparisons of party systems possible. **The Inverse Simpson index**,

$$1/\lambda = \frac{1}{\sum_{i=1}^{R} p_i^2} = {}^2D$$

Where λ equals the probability that two types taken at random from the dataset (with replacement) represent the same type. This simply equals true fragmentation of order 2, i.e. the effective number of parties that is obtained when the weighted arithmetic mean is used to quantify average proportional diversity of political parties in the election of interest.

Another measure is the **Least squares index (lsq)**, which measures the disproportionality produced by the election. Specifically, by the disparity between the distribution of votes and seats allocation.

Recently, Grigorii Golosov proposed a new method for computing the effective number of parties in which both larger and smaller parties are not attributed unrealistic scores as those resulted by using the Laakso/Taagepera index.I will call this as (**Golosov**) and is given by the following formula:

$$N = \sum_{i=1}^{n} \frac{p_i}{p_i + p_i^2 - p_i^2}$$

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

References

Gallagher, Michael and Paul Mitchell (2005) *The Politics of Electoral Systems.* Oxford University Press.

Golosov, Grigorii (2010) The Effective Number of Parties: A New Approach, *Party Politics,* **16:** 171-192.

Laakso, Markku and Rein Taagepera (1979) Effective Number of Parties: A Measure with Application to West Europe, *Comparative Political Studies,* **12:** 3-27.

Nicolau, Jairo (2008) *Sistemas Eleitorais.* Rio de Janeiro, FGV.

Taagepera, Rein and Matthew S. Shugart (1989) *Seats and Votes: The Effects and Determinants of Electoral Systems.* New Haven: Yale University Press.

See Also

cox.shugart, inv.cox.shugart, farina, grofman, gallagher, lijphart. For more details see the Indices vignette: `vignette("Indices", package = "SciencesPo")`

Examples

```
# Here are some examples, help yourself:
# The wikipedia examples

A <- c(.75,.25);
B <- c(.75,.10,rep(0.01,15))
C <- c(.55,.45);

# The index by "laakso/taagepera" is the default
politicalDiversity(A)
politicalDiversity(B)

# Using Grigorii Golosov method gives:
politicalDiversity(B, index="golosov")
politicalDiversity(C, index="golosov")

# The 1980 presidential election in the US (vote share):
US1980 <- c("Democratic"=0.410, "Republican"=0.507,
"Independent"=0.066, "Libertarian"=0.011, "Citizens"=0.003,
"Others"=0.003)

politicalDiversity(US1980)

politicalDiversity(US1980, index= "herfindahl")

politicalDiversity(US1980, index = "H") # will match Herfindahl

# The 1999 Finland election:
votes_1999 <- c(612963, 600592, 563835,
291675, 194846, 137330, 111835, 28084, 26440, 28549, 20442,
10378, 10104, 5451, 5194, 4481, 3903, 3455, 21734)

seats_1999 <- c(51, 48, 46, 20, 11, 11, 10, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)

# 2010 Brazilian legislative election

votes_2010 = c("PT"=13813587, "PMDB"=11692384, "PSDB"=9421347,
"DEM"=6932420, "PR"=7050274, "PP"=5987670, "PSB"=6553345,
"PDT"=4478736, "PTB"=3808646, "PSC"=2981714, "PV"=2886633,
"PC do B"=2545279, "PPS"=2376475, "PRB"=1659973, "PMN"=1026220,
"PT do B"=605768, "PSOL"=968475, "PHS"=719611, "PRTB"=283047,
"PRP"=232530, "PSL"=457490,"PTC"=563145)

seats_2010 = c("PT"=88, "PMDB"=79, "PSDB"=53, "DEM"=43,
"PR"=41, "PP"=41, "PSB"=34, "PDT"=28, "PTB"=21, "PSC"=17,
"PV"=15, "PC do B"=15, "PPS"=12, "PRB"=8, "PMN"=4, "PT do B"=3,
 "PSOL"=3, "PHS"=2, "PRTB"=2, "PRP"=2, "PSL"=1,"PTC"=1)

politicalDiversity(seats_2010)

politicalDiversity(seats_2010, index= "golosov")
```

`Presidents` *The Measure of US Presidents*

Description

The US presidents and their main opponents' heights (cm). This dataset contains the following columns:

- election The election year.
- winner The winner candidate.
- winner.height The winner candidate's height in centimeters (cm).
- winner.vote The popular vote support for the winner.
- winner.party The winner's party.
- opponent The main opponent candidate.
- opponent.height The opponent candidate's height in centimeters (cm).
- opponent.vote Popular vote support for the main opponent candidate.
- opponent.party The opponent's party.
- turnout The electorate turnout in percentages.
- winner.bmi The winner Body Mass Index (BMI) estimate (`BMI = weight in kg/(height in meter)**2`)

Usage

```
data(Presidents)
```

Format

A `data.frame` object with 11 variables and 57 observations.

Source

US Presidents: http://www.jimwegryn.com/Names/Presidents.php Inside Gov. http://www.us-presidents.insidegov.com. Wikipedia: http://en.wikipedia.org/wiki/United_States_presidential_election,_2012. Wikipedia:http://en.wikipedia.org/wiki/Heights_of_presidents_and_presidential_candidates_of_the_United_States.

---

psum — *The Missing R Parallel Sum*

Description

Provides parallel sum like `pmin` and `pmax` from the base package. The function `sum` simply does not help when the objective is to obtain a vector with parallel sum rather than a scalar value.

Usage

```
psum(..., na.rm = FALSE)
```

Arguments

`na.rm`	A logical value `TRUE` or `FALSE`, the default
`...`	One or more unit objects

Value

A vector containing the parallel sum.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
if (interactive()) {
n <- 20;
mydat <- data.frame(PT = rnorm(n, mean = .30),
PSDB = rnorm(n, mean = .25), PSB = rnorm(n, mean = .15));
head(mydat);
transform(mydat, DK = psum(PT, PSDB, PSB - 1));
}
```

---

pub_color_pal *Color Palettes for Publication (discrete)*

Description

Color palettes for publication-quality graphs. See details.

Usage

```
pub_color_pal(palette = "pub12")
```

Arguments

`palette`	Palette name.

Details

The following palettes are available:

- "pub12"Default colors of theme_pub
- "tableau20"Based on software Tableau
- "tableau10"Based on software Tableau
- "colorblind"Based on software Tableau
- "tableau10light"Based on software Tableau

Examples

```
library(scales)
show_col(pub_color_pal("pub12")(12))
show_col(pub_color_pal("tableau20")(20))
show_col(pub_color_pal("tableau10")(10))
show_col(pub_color_pal("colorblind")(10))
show_col(pub_color_pal("tableau10light")(10))
```

randomImput — *Simple Random Imputation*

Description

Performs random imputation in a vector containing missing values.

Usage

```
randomImput(x)
```

Arguments

x	a vector whose missing values (NA) is to be replaced.

Details

Indeed a very simple but somewhat limited approach is to impute missing values from observed ones chosen at random with replacement (MCAR), assuming that

$$p(R|Z_{obs}, Z_{mis}) = p(R|\phi)$$

. Sampling with replacement is important since it continues to favor values with higher incidence (preserving the MCAR empirical distribution). It may also be combined with apply for matrix imputation drills, but keep in mind that it is experimental (actually, I wrote this for teaching purposes).

Examples

```
x <- c(1,2,NA,4,5,NA)
randomImput(x)

if (interactive()) {
n = 100
mat <- matrix(ncol=3, nrow=n)
for(i in 1:n){
mu = mean(randomImput(x))
med = median(randomImput(x))
mod = Mode(randomImput(x))
mat[i,] <- c(mu, med, mod[1])
}
print(mat)
}
```

rdirichlet — *Dirichlet distribution*

Description

Density function and random number generation for the Dirichlet distribution

Usage

```
rdirichlet(n, alpha)
```

Arguments

n	number of random observations to draw.
alpha	the Dirichlet distribution's parameters. Can be a vector (one set of parameters for all observations) or a matrix (a different set of parameters for each observation), see "Details". If alpha is a matrix, a complete set of α-parameters must be supplied for each observation. log returns the logarithm of the densities (therefore the log-likelihood) and sum.up returns the product or sum and thereby the likelihood or log-likelihood.

Value

the rdirichlet returns a matrix with n rows, each containing a single random number according to the supplied alpha vector or matrix.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
# 1) General usage:
rdirichlet(20, c(1,1,1) );
alphas <- cbind(1:10, 5, 10:1);
alphas;
rdirichlet(10, alphas );
alpha.0 <- sum( alphas );
test <- rdirichlet(10, alphas );
apply( test, 2, mean );
alphas / alpha.0;
apply( test, 2, var );
alphas * ( alpha.0 - alphas ) / ( alpha.0^2 * ( alpha.0 + 1 ) );

# 2) A pratical example of usage:
# A Brazilian face-to-face poll by Datafolha conducted on Oct 03-04
# with 18,116 insterviews asking for their preferences for the
# presidential candidates.

## First, draw a sample from the posterior
set.seed(1234);
n <- 18116;
poll <- c(40,24,22,5,5,4) / 100 * n; # The data
mcmc <- 100000;
sim <- rdirichlet(mcmc, alpha = poll + 1);

## Second, look at the margins of Aecio over Marina in the very last minute of the campaign:
margin <- sim[,2] - sim[,3];
mn <- mean(margin); # Bayes estimate
mn;
s <- sd(margin); # posterior standard deviation

qnts <- quantile(margin, probs = c(0.025, 0.975)); # 90% credible interval
```

```
qnts;
pr <- mean(margin > 0); # posterior probability of a positive margin
pr;

## Third, plot the posterior density
hist(margin, prob = TRUE, # posterior distribution
  breaks = "FD", xlab = expression(p[2] - p[3]),
  main = expression(paste(bold("Posterior distribution of "), p[2] - p[3])));
abline(v=mn, col='red', lwd=3, lty=3);
```

`read.zTree` *Reads zTree output files*

Description

Extracts variables from a zTree output file.

Usage

```
read.zTree(object, tables = c("globals", "subjects"))
```

Arguments

`object`	a zTree file or a list of files.
`tables`	the tables of intrest.

Value

A list of dataframes, one for each table

Examples

```
## Not run: url <-
zTables <- read.zTree( "131126_0009.xls" , "contracts" )
zTables <- read.zTree( c("131126_0009.xls",
"131126_0010.xls"), c("globals","subjects", "contracts" ))

## End(Not run)
```

`recode` *Recode or Replace Values With New Values*

Description

Recodes a value or a vector of values.

Usage

```
recode(x, from, to, warn = TRUE)

## Default S3 method:
recode(x, from, to, warn = TRUE)
```

Arguments

x	The vector whose values will be recoded.
from	a vector of the items to recode.
to	a vector of replacement values.
warn	A logical to print a message if any of the old values are not actually present in x.

Examples

```
x <- LETTERS[1:5]
recode(x, c("B", "D"), c("Beta", "Delta"))

# On numeric vectors
x <- c(1, 4, 5, 9)
recode(x, from = c(1, 4, 5, 9), to = c(10, 40, 50, 90))
```

---

replicatedSampling *Replicated Simple Random Sample (SRS)*

Description

Replicated Simple Random Sample (SRS).

Usage

```
replicatedSampling(N = 500, n = 30, g = 6)
```

Arguments

N	The population size.
n	The sample size.
g	Number of independent sub-samples, each containing m = n/g units. Notice that m has to be a multiple of n and g.

Examples

```
replicatedSampling(500, 30, 6)
```

rosenbluth *Rosenbluth Index of Concentration*

Description

Calculates the Rosenbluth Index of concentration, also known as Hall or Tiedemann Indices.

Usage

```
rosenbluth(x, n = rep(1, length(x)), na.rm = FALSE, ...)

rosenbluth(x, n = rep(1, length(x)), na.rm = FALSE, ...)
```

Arguments

`x`	A vector of data values of non-negative elements.
`n`	A vector of frequencies of the same length as `x`.
`na.rm`	A logical. Should missing values be removed? The Default is set to `na.rm=FALSE`.
`...`	Additional arguements (currently ignored)

References

Cowell, F. A. (2000) Measurement of Inequality in Atkinson, A. B. / Bourguignon, F. (Eds): *Handbook of Income Distribution*. Amsterdam.

Cowell, F. A. (1995) *Measuring Inequality* Harvester Wheatshef: Prentice Hall.

See Also

`atkinson`, `herfindahl`, `gini`. For more details see the Indices vignette: `vignette("Indices", package = "Sciences`

Examples

```
# generate a vector (of incomes)
x <- c(778, 815, 857, 888, 925, 930, 965, 990, 1012)

# compute Rosenbluth coefficient
rosenbluth(x)
```

rounded *Round Numbers Without Leading Zeros*

Description

Given a numeric vector, round numbers with no leading zeros. Something nice for a plot or publicatio.

Usage

```
rounded(x, digits = 2, add = FALSE, max = (digits + 2))
```

Arguments

x	A numeric vector.
digits	An integer for the number of digits to round to.
add	Logical, whether additional digits are to be added if no number appears in the pre-set digit level, default is FALSE.
max	The Maximum number of digits to be shown, only affects if add=TRUE.

Value

A vector of the same length of x, but stored as string.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

Examples

```
x = seq(0, 1, by=.1)
rounded(x)
```

---

runExample *Examples from the SciencesPo Package*

Description

Launchs a Shiny app that shows a demo.

Usage

```
runExample(example)
```

Arguments

example	The name of the shiny application.

Examples

```
# A demo of what \code{\link[SciencesPo]{politicalDiversity}} does.
if (interactive()) {
runExample(politicalDiversity)
}
```

safe.chars — *Convert All Factor Columns to Character Columns*

Description

By default, R converts character columns to factors. Instead of re-reading the data using `stringsAsFactors`, the `safe.chars` function will identify which columns are currently factors, and convert them all to characters.

Usage

```
safe.chars(.data)
```

Arguments

`.data`	The name of the `data.frame`

Author(s)

Daniel Marcelino, <dmarcelino@live.com>.

See Also

`read.table`, `destring`.

Examples

```
str(iris)
iris_2 = safe.chars(iris)
str(iris_2)
```

samplePower — *Calculate and plot power of a sample*

Description

Calculates and plots power of a sample z-test of a sample mean mu1 against a population mean `mu0` (H_0: mu0 = mu1, H_1: mu0 <> mu1).

Usage

```
samplePower(mu0 = 0, mu1 = 0, sigma = 1, n = 100, alpha = 0.05)
```

Arguments

`mu0`	This should be the "known" mean value for your population.
`mu1`	This should be the "expected" mean value from your sample. The delta between mu(0) and mu(1) is what you should consider a significant difference for the test.
`sigma`	This should be the known sigma (standard deviation) for the population.
`n`	The sample size.
`alpha`	This is the significance level, default is alpha(twosided) = .05.

Details

sample.power calculates the power of a one-sample z-test (twosided) and plots the density distributions under the assumption of of H_0: m = mu0 and H_1: m = mu1. The rejection regions of H_0 (alpha) are colored blue, while the rejection region of H_1 (beta) is colored red.

Value

n the sample size; sigma the standard deviation; SE the standard error of the mean; mu0 the mean of H_0 in the population; mu1 the sample mean; mean.crit the critical value of sample mean to achieve significance; ES the population "effect" size gamma; delta the effect size delta (Cohen); alpha the significance level alpha (twosided); power the power (1-beta).

Examples

```
samplePower(mu0=68, mu1=69, sigma=3.1, n=100)
## gives a power of .90
```

---

sampleSize *Simple Sample Size for Surveys*

Description

Compute sample size for surveys.

Usage

```
sampleSize(p, delta = "auto", popsize = NULL, deff = 1, alpha = 0.05)
```

Arguments

p	The proportion.
delta	The error size.
popsize	An integer for the population size.
deff	An intger for the deff.
alpha	The level of alpha/significance.

Examples

```
# Comercial public opinion samples in Brazil:
sampleSize(p=.50, delta=.03)
sampleSize(p=.50, delta=.02)
```

scale_colour_fte — *fivethirtyeight.com color scales*

Description

Color scales using the colors in the fivethirtyeight graphics.

Usage

```
scale_colour_fte(...)

scale_color_fte(...)

scale_fill_fte(...)
```

Arguments

`...`	Other arguments passed on to `discrete_scale` to control name, limits, breaks, labels and so forth.

See Also

`theme_538` for examples.

Other colour fte: `fte_color_pal`

scale_colour_parties — *Political Parties Color Scales*

Description

Scale color for political parties.

Usage

```
scale_colour_parties(palette = "BRA", ...)

scale_fill_parties(palette = "BRA", ...)

scale_color_parties(palette = "BRA", ...)
```

Arguments

`palette`	Palette name.
`...`	Other arguments passed on to `discrete_scale` to control name, limits, breaks, labels and so forth.

See Also

`parties_color_pal` for references.

Other colour parties: `parties_color_pal`

scale_colour_pub *Publication color scales.*

Description

See pub_color_pal for details.

Usage

```
scale_colour_pub(palette = "tableau20", ...)

scale_fill_pub(palette = "tableau20", ...)

scale_color_pub(palette = "tableau20", ...)
```

Arguments

palette	Palette name.
...	Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.

See Also

pub_color_pal for references.

SciencesPo *A Tool Set For Analyzing Political Behavior Data*

Description

Provides functions for analyzing political behavior data, including measures of political fragmentation, seats allocation, and graphical demonstrations.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Marcelino, Daniel (2013). *SciencesPo: A Tool Set for Analyzing Political Behaviour Data*. Available at SSRN: http://dx.doi.org/10.2139/ssrn.2320547

`se` *Calculates the Standard Error of the Mean*

Description

Computes the standard error of the sample mean.

Usage

```
se(x, na.rm = TRUE, ...)

## Default S3 method:
se(x, na.rm = TRUE, ...)

## S3 method for class 'data.frame'
se(x, na.rm = TRUE, ...)
```

Arguments

`x`	An R object.
`na.rm`	A logical value indicating whether `NA` should be stripped before the computation proceeds. Default is `na.rm=TRUE`.
`...`	Additional arguements (currently ignored)

Details

The standard error of the mean (SEM) (*assuming statistical independence of the values in the sample*) is estimated by taking the standard deviation of the population sample, divided by the square root of the sample size:

$$se = \frac{s}{\sqrt{n}}$$

Author(s)

Daniel Marcelino, `<dmarcelino@live.com>`

Examples

```
x <- c(1, 2.3, 2, 3, 4, 8, 12, 43, -1,-4)
myse <- sd(x)/sqrt(length(x))
myse
# With the 'se' function:
se(x)
```

sheston91 *The Penn World Table*

Description

The Penn World Table used in Summers and Heston (1991). This dataset contains the following columns:

- year Year
- pop Population (thousands)
- rgdppc Real per capita GDP
- savrat a numeric vector
- country Country
- com Communist regime
- opec OPEC country
- name Country name

Usage

```
data(sheston91)
```

Format

A `data.frame` object with 8 variables and 3250 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Summers, R. and Heston, A. (1991) The Penn World Table (Mark 5): an expanded set of international comparisons, 1950–1988. *The Quarterly Journal of Economics,* **106(2),** 327–368.

skewness *Compute the Skewness*

Description

The function provides three features to perform a skewness test, see details below.

Usage

```
skewness(x, na.rm = TRUE, type = 2)
```

Arguments

- `x` — a numeric vector containing the values whose skewness is to be computed.
- `na.rm` — a logical value for `na.rm`, default is `na.rm=TRUE`.
- `type` — an integer between 1 and 3 for selecting the algorithms for computing the skewness, see details below.

Details

The skewness is a measure of symmetry distribution. Intuitively, negative skewness (g_1 < 0) indicates that the mean of the data distribution is less than the median, and the data distribution is left-skewed. Positive skewness (g_1 > 0) indicates that the mean of the data values is larger than the median, and the data distribution is right-skewed. Values of g_1 near zero indicate a symmetric distribution. The skewness function will ignore missing values in 'x' for its computation purpose. There are several methods to compute skewness, Joanes and Gill (1998) discuss three of the most traditional methods. According to them, **type 3** performs better in non-normal population distribution, whereas in normal-like population distribution type 2 fits better the data. Such difference between the two formulae tend to disappear in large samples. **Type 1:** g_1 = m_3/m_2^(3/2).

Type 2: G_1 = g_1*sqrt(n(n-1))/(n-2).

Type 3: b_1 = m_3/s^3 = g_1 ((n-1)/n)^(3/2).

Value

An object of the same type as `x`

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Joanes, D. N. and C. A. Gill. (1998) Comparing measures of sample skewness and kurtosis. *The Statistician,* **47,** 183–189.

Examples

```
w <-sample(4,10, TRUE)
x <- sample(10, 1000, replace=TRUE, prob=w)
skewness(x, type = 1)
skewness(x)
skewness(x, type = 3)
```

`star` *Returns significance level.*

Description

Returns the significance level as stars, or NA if a non-numeric value is passed in.

Usage

```
star(x)
```

Arguments

x	p-value.

---

stdkurtosis *Standard Error of Kurtosis*

Description

Generate the standard error of the kurtosis.

Usage

```
stdkurtosis(x, na.rm = TRUE)

## Default S3 method:
stdkurtosis(x, na.rm = TRUE)

## S3 method for class 'data.frame'
stdkurtosis(x, na.rm = TRUE)
```

Arguments

x	An R object.
na.rm	a logical value indicating whether NA should be stripped before the computation proceeds.

---

stdskewness *Standard Error of Skewness*

Description

Generate the standard error of the skewness.

Usage

```
stdskewness(x, na.rm = TRUE)

## Default S3 method:
stdskewness(x, na.rm = TRUE)

## S3 method for class 'data.frame'
stdskewness(x, na.rm = TRUE)
```

Arguments

x	An R object.
na.rm	a logical value indicating whether NA should be stripped before the computation proceeds.

stratified *Stratified Sampling*

Description

A handy function for sampling row values of a data.frame conditional to some strata.

Usage

```
stratified(.data, group, size, select = NULL, replace = FALSE,
  both.sets = FALSE)
```

Arguments

.data	The data.frame from which the sample is desired.
group	The grouping factor, may be a list.
size	The sample size.
select	If sampling from a specific group or list of groups.
replace	Should sampling be with replacement?
both.sets	If TRUE, both 'sample' and '.data' are returned.

Examples

```
# Generate a couple of sample data.frames to play with

set.seed(51)
dat1 <- data.frame(ID = 1:100, A = sample(c("AA", "BB", "CC", "DD", "EE"),
100, replace = TRUE), B = rnorm(100), C = abs(round(rnorm(100), digits = 1)),
D = sample(c("CA", "NY", "TX"), 100, replace = TRUE), E = sample(c("M","F"),
100, replace = TRUE))

# Let's take a 10% sample from all -A- groups in dat1
 stratified(dat1, "A", 0.1)

 # Let's take a 10% sample from only 'AA' and 'BB' groups from -A- in dat1
 stratified(dat1, "A", 0.1, select = list(A = c("AA", "BB")))

 # Let's take 5 samples from all -D- groups in dat1, specified by column
stratified(dat1, group = 5, size = 5)

# Let's take a sample from all -A- groups in dat1, where we specify the
# number wanted from each group
stratified(dat1, "A", size = c(3, 5, 4, 5, 2))

# Use a two-column strata (-E- and -D-) but only interested in cases where
# -E- == 'M'
stratified(dat1, c("E", "D"), 0.15, select = list(E = "M"))
```

stukel *Stukel's test of the logistic link*

Description

The Stukel's test is an alternative to the goodness-of-fit test for logistic regression. It tests if significant change occurs in the model with the addition of new coefficients.

Usage

```
stukel(object, alternative = c("both", "alpha1", "alpha2"))
```

Arguments

`object`	An object of class `glm`.
`alternative`	add both `z1` and `z2` to model or just one of them.

Details

Two new covariates, z1 and z2 are generated such that

$$z1 = 0.5 logit^2 * I(pi >= 0.5)$$

,

$$z2 = -0.5 logit^2 I(pi <= 0.5)$$

, where

$$I(arg) = 1$$

if arg is `TRUE` and

$$I(arg) = 1$$

if `FALSE`.

Note

Adapted from program published by Brett Presnell's code available at the Florida University.

References

Stukel, T.A. (1988) Generalized logistic models. *Journal of the American Statistical Association* 83: 426-431.

Hosmer, David W., et al (1997) A comparison of goodness-of-fit tests for the logistic regression model. *Statistics in medicine* 16.9, 965-980.

Allison, Paul (2014) *Another Goodness-of-Fit Test for Logistic Regression*.

sturges — *Calculate breaks*

Description

Calculate breaks according to the Herbert Sturges' (1926) formula

Usage

```
sturges(x)
```

Arguments

x	A vector of count values.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Sturges, H. (1926) *The choice of a class-interval.* J. Amer. Statist. Assoc., 21, 65-66.

Examples

```
set.seed(51)
y <- sample(100)
sturges(y)
```

svTransform — *Transform dependent variable*

Description

Simple function to transform a dependent variable that in [0,1] rather than (0, 1) to beta regression. Suggested by Smithson & Verkuilen (2006).

Usage

```
svTransform(y)
```

Arguments

y	the dependent variable in [0, 1] interval.

References

Smithson M, Verkuilen J (2006) A Better Lemon Squeezer? Maximum-Likelihood Regression with Beta-Distributed Dependent Variables. *Psychological Methods*, 11(1), 54-71.

See Also

normalize.

Examples

```
x <- sample(10); x;
 y <- normalize(x); y;
 svTransform(y)
```

---

swatson93 *Stock's and Watson's (1993) Data.*

Description

Data set used by Stock and Watson (1993) to estimate co-integration. This dataset contains the following columns:

- lnm1 Log M1.
- lnp Log NNP price deflator.
- lnnnp Log NNP.
- cprate A numeric vector.
- year Commercial paper rate.

Usage

```
data(swatson93)
```

Format

A data.frame object with 5 variables and 90 observations.

Source

Hayashi, F. (2000). *Econometrics.* Princeton. New Jersey, USA: Princeton University. http://fmwww.bc.edu/ec-p/data/Hayashi/

References

Stock, J. H., and Watson, M. W. (1993) A simple estimator of cointegrating vectors in higher order integrated systems. *Econometrica: Journal of the Econometric Society,* 783–820.

`tabstat` *Table With Summary Statistics*

Description

Generates a table with summary statistics.

Usage

```
tabstat(.data, var, by = NULL, statistics = c("nnmiss", "mean", "sd"))

## Default S3 method:
tabstat(.data, var, by = NULL, statistics = c("nnmiss",
  "mean", "sd"))
```

Arguments

`.data`	The `data.frame` names.
`var`	The name of variable or column.
`by`	The name of grouping variable.
`statistics`	The name of desired statistics.

Examples

```
tab = tabstat(Presidents, c("winner.height", "winner.vote", "turnout"))
# knitr::kable(tab, digits=2)
```

`textwrap` *Insert line breaks in long strings*

Description

Insert line breaks in long character strings. Useful for wrapping labels / titles for plots and tables.

Usage

```
textwrap(labels, wrap, linesep = NULL)
```

Arguments

`labels`	Label(s) as character string, where a line break should be inserted. Several strings may be passed as vector (see 'Examples').
`wrap`	Maximum amount of chars per line (i.e. line length). If codewrap = Inf, no word wrap will be performed (i.e. `labels` will be returned as is).
`linesep`	By default, this argument is `NULL` and a regular new line string (`"\n"`) is used. For HTML-purposes, for instance, `linesep` could be `" "`.

Value

New label(s) with line breaks inserted at every wrap's position.

Examples

```
textwrap(c("A very long string", "And another even longer string!"), 10)
```

---

themes_data *Palette data for the themes used by package*

Description

Data used by the palettes in the package.

Usage

```
themes_data
```

Format

A list.

---

theme_blank *Create a Completely Empty Theme*

Description

The theme created by this function shows nothing but the plot panel.

Usage

```
theme_blank(font_size = 12, font_family = "")
```

Arguments

font_size	Overall font size. Default is 12.
font_family	Default font family.

Value

The theme.

Examples

```
# plot with small amount of remaining padding
qplot(1:10, (1:10)^2) + theme_blank()
# remaining padding removed
qplot(1:10, (1:10)^2) + theme_blank() + labs(x = NULL, y = NULL)
```

theme_fte *Themes for ggplot2 Graphs*

Description

Theme for plotting with ggplot2.

Usage

```
theme_fte(legend = "none", font_size = 12, font_family = "sans",
  colors = c("#F0F0F0", "#D0D0D0", "#535353", "#3C3C3C"))

theme_538(legend = "none", font_size = 12, font_family = "sans",
  colors = c("#F0F0F0", "#D0D0D0", "#535353", "#3C3C3C"))
```

Arguments

legend	Enables to set legend position, default is "none".
font_size	Overall font size. Default is 13.
font_family	Default font family.
colors	Default colors used in the plot in the following order: background, lines, text, and title.

Value

The theme.

Examples

```
qplot(1:10, (1:10)^3) + theme_fte()
```

theme_pub *The Default Theme*

Description

After loading the SciencesPo package, this theme will be set to default for all subsequent graphs made with ggplot2.

Usage

```
theme_pub(legend = "bottom", font_family = "sans", font_size = 13,
  line_width = 0.5, axis.line.x = element_line(),
  axis.line.y = element_blank())
```

Arguments

legend	Enables to set legend position, default is "bottom".
font_family	Default font family.
font_size	Overall font size. Default is 14.
line_width	Default line size.
axis.line.x	Enables to set x axis line.
axis.line.y	Enables to set y axis line.

Value

The theme.

See Also

theme, theme_538, theme_blank.

Examples

```
ggplot(diamonds,aes(cut, group=1)) + geom_bar()+
geom_freqpoly(stat="count",size=2) + scale_color_pub() + theme_pub(line_width=1)

dat <- data.frame()
for(i in 1:4)
dat <- rbind(dat, data.frame(set=i, x=anscombe[,i], y=anscombe[,i+4]))

ggplot(dat, aes(x, y)) + geom_point(size=5, color="red",
fill="orange", shape=21) + geom_smooth(method="lm", fill=NA,
fullrange=TRUE) + facet_wrap(~set, ncol=2)
```

---

timeplot — *Make the ggplot2 version of TS plots*

Description

The function produces TS plots using ggplot.

Usage

```
timeplot(ts, ylab = "", ylim = c(-1, 1), ci = 0.95, ...)
```

Arguments

ts	The TS object.
ylab	The y-axis title.
ylim	The y-axis limits.
ci	The desired confidence interval.
...	Ignored parameters passed to ggplot.

Examples

```
ts.sim <- stats::arima.sim(n = 100, list(ma=0.8), innov=rnorm(100))
 timeplot(ts.sim)
```

`titanic` *Titanic*

Description

Population at Risk and Death Rates for an Unusual Episode. For each person on board the fatal maiden voyage of the ocean liner Titanic, this dataset records sex, age [adult/child], economic status [first/second/third class, or crew] and whether or not that person survived. This dataset contains the following columns:

- CLASS Class (0 = crew, 1 = first, 2 = second, 3 = third)
- AGE Age (1 = adult, 0 = child)
- SEX Sex (1 = male, 0 = female)
- SURVIVED Survived (1 = yes, 0 = no)

Usage

```
data(titanic)
```

Format

A `data.frame` object with 4 variables and 2201 observations.

Note

There is not complete agreement among primary sources as to the exact numbers on board, rescued, or lost. **STORY BEHIND THE DATA:** The sinking of the Titanic is a famous event, and new books are still being published about it. Many well-known facts–from the proportions of first-class passengers to the "women and children first" policy, and the fact that that policy was not entirely successful in saving the women and children in the third class–are reflected in the survival rates for various classes of passenger. These data were originally collected by the British Board of Trade in their investigation of the sinking.

Source

British Board of Trade Inquiry Report (1990). *Report on the Loss of the 'Titanic' (S.S.)"*, Gloucester, UK: Allan Sutton Publishing.

References

Dawson (1995). "The 'Unusual Episode' Data Revisited" in the *Journal of Statistics Education*.

trim *Trim white spaces*

Description

Simply trims spaces from the start, end, and within of a string

Usage

```
trim(x, delim = " ")
```

Arguments

x	is a character vector.
delim	is the delimiter, default is white spaces " "

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

turnout *Turnout Data*

Description

Data on voter turnout in the 50 states and D.C. for the 1992 Presidential election and 1990 Congressional elections. Per capita income, populations in poverty and populations with no high school degree are also given. This dataset contains the following columns:

- v1 state name (alphabetic, 20 characters)
- v2 region of country: 1 Northeast 2 Midwest 3 South 4 West
- v3 division within region: 1 Northeast-New England 2 Northeast-Middle Atlantic 3 Midwest-East North Central 4 Midwest-West North Central 5 South-South Atlantic 6 South-East South Central 7 South-West South Central 8 West-Mountain 9 West-Pacific.
- v4 "Elazar's state political culture assignments: 1 = moralistic 2 = individualistic 3 = traditionalistic
- v5 percent of population below poverty level, 1992.
- v6 per capita personal income, 1993.
- v7 percent casting votes for presidential electors, 1992.
- v8 percent casting votes for U.S. Representatives, 1990.
- v9 population without a high school degree, of those 25 years or older, 1990.
- v10 population 25 years or older, 1990.
- v11 South =1; all others =0.

Usage

```
data(turnout)
```

Format

A `data.frame` object with 11 variables and 50 observations.

Source

U.S. Bureau of the Census, Statistical Abstract of the United States, 1994.

`twins` *Burt's twin data*

Description

The given data are IQ scores from identical twins; one raised in a foster home, and the other raised by birth parents. This dataset contains the following columns:

- C Social class, C1=high, C2=medium, C3=low, a factor.
- IQb biological.
- IQf foster.

Usage

```
data(twins)
```

Format

A `data.frame` object with 3 variables and 27 observations.

Source

Burt, C. (1966). The genetic estimation of differences in intelligence: A study of monozygotic twins reared together and apart. Br. J. Psych., 57, 147-153.

References

Weisberg, S. (2014). Applied Linear Regression, 4th edition. Hoboken NJ: Wiley.

`ucss` *Uncorrected Sum of Squares*

Description

Generate the uncorrected sum of squares.

Usage

```
ucss(x, na.rm = TRUE)

## Default S3 method:
ucss(x, na.rm = TRUE)

## S3 method for class 'data.frame'
ucss(x, na.rm = TRUE)
```

Arguments

x	An R object.
na.rm	A logical value indicating whether NA should be stripped before the computation proceeds.

Units *Measurement System Units*

Description

A dataset with measurement system units.

- from A character defining the original unit.
- to A character defining the target unit.
- factor The factor to be applied in conversion.
- description Some details about the measure.

Usage

```
data(Units)
```

Format

A data.frame object with 4 variables and 46 observations.

---

untable *Untable*

Description

Method for recreate the data.frame out of a contingency table, i.e., converts from summarized data to long.

Usage

```
untable(x, ...)

## S3 method for class 'data.frame'
untable(x, freq = "Freq", rownames = NULL, ...)

## Default S3 method:
untable(x, dimnames = NULL, type = NULL,
  rownames = NULL, colnames = NULL, ...)
```

Arguments

`x`	The table object as a data.frame, table, or, matrix.
`freq`	The column with count values.
`rownames`	Row names to add to the data.frame.
`dimnames`	Set dimnames of an object if require.
`type`	The type of variable. If NULL, ordered factor is returned.
`colnames`	Column names to add to the data.frame.
`...`	Extra parameters.

Examples

```
if (interactive()) {
gss <- data.frame(
expand.grid(sex=c("female", "male"),
party=c("dem", "indep", "rep")),
count=c(279,165,73,47,225,191))

print(gss)

# Then expand it:
GSS <- untable(gss, freq="count")
head(GSS)
}
```

---

vif — *Variance Inflation Factor*

Description

Extracts Variance Inflation Factor from a model of class “lm”

Usage

```
vif(model, ...)
```

Arguments

`model`	a model object
`...`	further arguments passed to or used by other methods.

Value

A numeric value indicating the variance inflation in the model

Author(s)

Daniel Marcelino, `<dmarcelino@live.com>` #’

Examples

```
data(mtcars)

m1 <- lm(mpg ~ qsec + hp, data=mtcars)

vif(m1)
```

`voronoi` *Voronoi diagram*

Description

Computes the voronoi diagram.

Usage

```
voronoi(p, n = 100, dim = 1000, plot = TRUE)
```

Arguments

`p`	An integer for the size of the
`n`	An integer for the size of
`dim`	The dimension of the image.
`plot`	Logical, if `TRUE`, the plot is returned, else, the `data.frame` is returned.

Details

https://en.wikipedia.org/wiki/Voronoi_diagram

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

Examples

```
## Not run:  voronoi(p=2, n=20, dim=1000)
```

`winsorize` *Winsorized Mean*

Description

Compute the winsorized mean, which consists of recoding the top k values in a vector.

Usage

```
winsorize(x, k = 1, na.rm = TRUE)
```

Arguments

`x`	The vector to be winsorized
`k`	An integer for the quantity of outlier elements that to be replaced in the calculation process
`na.rm`	a logical value for `na.rm`, default is `na.rm=TRUE`.

Details

Winsorizing a vector will produce different results than trimming it. While by trimming a vector causes extreme values to be discarded, by winsorizing it in the other hand, causes extreme values to be replaced by certain percentiles.

Value

An object of the same type as `x`

Note

One may want to winsorize estimators, however, winsorization tends to be used for one-variable situations.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

References

Dixon, W. J., and Yuen, K. K. (1999) Trimming and winsorization: A review. *The American Statistician,* **53(3),** 267–269.

Dixon, W. J., and Yuen, K. K. (1960) Simplified Estimation from Censored Normal Samples, *The Annals of Mathematical Statistics,* **31,** 385–391.

Wilcox, R. R. (2012) *Introduction to robust estimation and hypothesis testing.* Academic Press, 30-32. Statistics Canada (2010) *Survey Methods and Practices.*

Examples

```
set.seed(51)  # for reproducibility
x <- rnorm(50)
## introduce outliers
x[1] <- x[1] * 10
# Compare to mean:
 mean(x)
 winsorize(x)
```

---

words *Word frequencies from Mosteller and Wallace*

Description

The data give the frequencies of words in works from four different sources: the political writings of eighteenth century American political figures Alexander Hamilton, James Madison, and John Jay, and the book Ulysses by twentieth century Irish writer James Joyce. This dataset contains the following columns:

- Hamilton Hamilton frequency.
- HamiltonRank Hamilton rank.
- Madison Madison frequency.
- MadisonRank Madison rank.
- Jay Jay frequency.
- JayRank Jay rank.
- Ulysses Word frequency in Ulysses.
- UlyssesRank Word rank in Ulysses.

Usage

```
data(words)
```

Format

A `data.frame` object with 8 variables and 165 observations.

Source

Mosteller, F. and Wallace, D. (1964). Inference and Disputed Authorship: The Federalist. Reading, MA: Addison-Wesley.

References

Weisberg, S. (2014). Applied Linear Regression, 4th edition. Hoboken NJ: Wiley.

`wtd.var` *Weighted Variance*

Description

Weighted Variance Formula

Usage

```
wtd.var(x, w, na.rm = FALSE)
```

Arguments

x	the variable.
w	the variance.
na.rm	A logical if NA should be disregarded.

Examples

```
wt=c(1.23, 2.12, 1.23, 0.32, 1.53, 0.59, 0.94, 0.94, 0.84, 0.73)
x = c(5, 5, 4, 4, 3, 4, 3, 2, 2, 1)
wtd.var(x, wt)
```

`%>%` *Chain operator*

Description

Chain operator.

Usage

```
x
```

Index