Package 'basr'

February 6, 2014

Version 0.7.2

Date 2014-02-06

Title Basic, but hopefully useful, functions

Description This package provides a bunch of basic functions for a variety of usage.

Author Mathieu Basille, contributions from Samuel Brown, Marc in the box,Clement Calenge, Jean Lobry, Kevin Wright

Maintainer Mathieu Basille <basille@ase-research.org>

Suggests devtools

License GPL (>= 3)

URL http://ase-research.org/basille/basr

R topics documented:

`basr` *Utility functions*

Description

basr package

Details

This package provides a bunch of basic, but hopefully useful, functions for a variety of usage. For a list of documented functions, use `library(help = "basr")`

Author(s)

Mathieu Basille <basille@ase-research.org>, contributions from Samuel Brown, Marc in the box, Jean Lobry, Kevin Wright

---

`capwords` *Capitalizing*

Description

Capitalizing - every first letter of a word is changed to upper case.

Usage

```
capwords(s, strict = FALSE)
```

Arguments

`s`	A character vector, or an object that can be coerced to character by `as.character`.
`strict`	Logical: other letters than the first are converted to lower case

Value

A character vector of the same length and with the same attributes as `x` (after possible coercion).

Author(s)

From the help page of `chartr`

Examples

```
capwords(c("using AIC for model selection"))
## ->  [1] "Using AIC For Model Selection"
capwords(c("using AIC", "for MODEL selection"), strict = TRUE)
## ->  [1] "Using Aic"  "For Model Selection"
##                ^^^        ^^^^^
##              'bad'       'good'
```

`confint` *Confidence Intervals for Model Parameters*

Description

Modified version of the `confint` function, which displays the coefficients in addition to the CIs, and allows for more control on display parameters. A `plot` argument and function allow to graph the coefficients and their CIs.

Usage

```
confint(object, parm, level = 0.95, order = FALSE, groups, plot = FALSE,
  ...)

## S3 method for class 'confint'
plot(x, mar = c(5, 7, 3, 1) + 0.1, col = NULL,
  main = attr(x, "model"), pch = 19, ...)
```

Arguments

`order`	Logical. If `TRUE`, the results are ordered by descending order on the coefficient value.
`groups`	A `factor` in the sense that `as.factor(f)` defines the groups, or a list of such factors in which case their interaction is used for the groups. See `split`.
`plot`	Whether to plot the results.
`...`	Further arguments passed to `points`.
`x`	A `confint` object.
`mar`	The number of lines of margin, can be useful if the coefficient names do not fit in the left margin. See `par` for more details.
`col`	The color of each coefficient + CI; gray by default. If `"groups"`, the color of each (sorted) group; use a hcl palette by default.

Value

A data frame providing the CI and coefficients.

Author(s)

Mathieu Basille <basille@ase-research.org>

See Also

confint for more details on other parameters.

Examples

```
## Example of linear model
fit <- lm(100/mpg ~ disp + hp + wt + am, data = mtcars)
## Standard 'confint' function
stats::confint(fit)
## Same results with modified function
confint(fit)
## Argument 'level'
stats::confint(fit, level = .9)
confint(fit, level = .9)
## Argument 'order'
confint(fit, order = TRUE)
## Argument 'groups'
confint(fit, groups = c(3, 1, 1, 1, 2))
## Argument 'level', "'order' and 'groups' simultaneously
confint(fit, level = .9, order = TRUE, groups = c(3, 1, 1, 1, 2))
## Argument 'parm'
stats::confint(fit, "am")
confint(fit, "am")

## Plot of the results
plot(confint(fit, order = TRUE, groups = c(3, 1, 1, 1, 2)))
confint(fit, order = TRUE, groups = c(3, 1, 1, 1, 2), plot = TRUE)
confint(fit, order = TRUE, groups = c(3, 1, 1, 1, 2), plot = TRUE,
    col = c("blue", "red", "green"), pch = 18, cex = 2)
confint(fit, order = TRUE, groups = c(3, 1, 1, 1, 2), level = 0.9,
    plot = TRUE)
```

---

cv *Coefficient of variation*

Description

This function computes the coefficient of variation (i.e. sd / mean) of the values in x. If ci is TRUE then confidence intervals are also computed.

Usage

```
cv(x, na.rm = FALSE, ci = FALSE, conf.level = 0.95,
  method = c("mckaymod", "mckay", "naive"))
```

Arguments

x	A numeric vector
na.rm	Logical. Should missing values be removed?
ci	Logical. Should confidence intervals be computed?
conf.level	Confidence level of the interval.
method	The method to compute the confidence intervale. Either the naive (naive), the McKay (mckay) or the modified McKay (mckaymod, default) approximation.

Value

If ci, returns a list with the coefficient of variation. in the first element and the confidence interval in the second.

Original URL

http://tolstoy.newcastle.edu.au/R/e2/help/07/06/19043.html

Author(s)

From Kevin Wright, modified by Mathieu Basille <basille@ase-research.org>

References

Vangel, M. G. (1996) Confidence intervals for a normal coefficient of variation. The American Statistician, 50: 21-26

Examples

```
xx <- 1:10
cv(xx)
sd(xx)/mean(xx)
cv(xx, ci = TRUE)
```

---

dynamitePlot *Dynamite Plots*

Description

Creates dynamite plots.

Usage

```
dynamitePlot(height, error, names.arg = NULL, significance = NA,
  ylim = c(0, maxLim), sym = FALSE, head = 0.7, lwd = par("lwd"),
  cex.sig = 1.2, ...)
```

Arguments

height	A vector of values describing the heights of the rectangular bars which make up the plot.
error	A vector of values indicating the length of error bars.
names.arg	A vector of names to be plotted below each bar or group of bars. If this argument is omitted, then the names are taken from the names attribute of height.
significance	A character vector giving the group significance for each value.
ylim	Limits for the y axis. By default, ylim uses c(0, maxLim), where maxLim is the maximum height + error multiplied by a factor of 1.1.
sym	Logical. Whether to draw lower error bars.
head	A numeric, which gives the approximate width of the head, relative to the bar width.
lwd	The line width of the error bars, a _positive_ number, defaulting to par("lwd") (usually 1).
cex.sig	The magnification to be used for significance groups relative to the current setting of cex (which defaults to 1).
...	Arguments to be passed to barplot.

Original URL

http://the-praise-of-insects.blogspot.ca/2012/04/dynamite-plots-in-r.html

Note

Ben Bolker wrote an extensive discussion of the advantages and disadvantages of dynamite plots here: http://emdbolker.wikidot.com/blog:dynamite

Author(s)

Samuel Brown, modified by Mathieu Basille <basille@ase-research.org>

Examples

```
values <- c(1, 2, 5, 4)
errors <- c(0.25, 0.5, 0.33, 0.12)
names <- paste("Trial", 1:4)
sig <- c("a", "a", "b", "b")
dynamitePlot(values, errors)
par(mar = c(3, 5, 1, 1) + .1)
dynamitePlot(values, errors, names.arg = names, significance = sig,
    ylab = "Values", sym = TRUE, cex.lab = 1.5, cex.axis = 1.2,
    cex.names = 1.2, cex.sig = 1.5, space = c(0, 0.2, 0.8, 0.2),
    lwd = 2, head = 0, col = c(grey(0.5), "white"), border = c(NA,
        "black"))
```

extrange *Extended range*

Description

Returns the range extended by a given proportion.

Usage

```
extrange(x, percent = 0.1, na.rm = FALSE)
```

Arguments

x	A numeric vector.
percent	The proportion to be added to the range.
na.rm	Logical, indicating if NA's should be omitted.

Details

If the regular range returns a single value, the proportion is computed on this value itself (and not on the range).

Author(s)

Mathieu Basille <basille@ase-research.org>

Examples

```
extrange(0:10)
extrange(0:10, percent = .5)
extrange(-10:10)
extrange(rep(10, 3))
```

getcolors *Choosing colors visually*

Description

Allows for the selection of n colors by using a simplified color swatch.

Usage

```
getcolors(n)
```

Arguments

n	The number of colors to choose

Details

getcolors allows selection with a mouse using the locator function. Following selection, a second plot opens showing how these colors look next to each other and on a background gradient of black to white. The function uses an RGB color model: Red increases on the y-axis, Green increases on the x-axis, and Blue is a repeated sequence of levels across the x-axis.

Value

A character vector with elements of 7 or 9 characters, "#" followed by the red, blue, green and optionally alpha values in hexadecimal (after rescaling to 0 ... 255). The optional alpha values range from 0 (fully transparent) to 255 (opaque).

Original URL

http://menugget.blogspot.com/2013/01/choosing-colors-visually-with-getcolors.html

Author(s)

Marc in the box

Examples

```
## Not run:
set.seed(1)
n <- 100
x <- seq(n)
y1 <- cumsum(rnorm(n))
y2 <- cumsum(rnorm(n))
y3 <- cumsum(rnorm(n))
y4 <- cumsum(rnorm(n))
ylim <- range(c(y1, y2, y3, y4))

cols <- getcolors(4)

plot(x, y1, ylim = ylim, t = "l", col = cols[1], lwd = 3, ylab = "")
lines(x, y2, col = cols[2], lwd = 3)
lines(x, y3, col = cols[3], lwd = 3)
lines(x, y4, col = cols[4], lwd = 3)
legend("topleft", legend = paste("y", 1:4, sep = ""), col = cols,
    lwd = 3)
## End(Not run)
```

---

manual *Generate package reference manual*

Description

Generate package reference manual. This function requires the devtools package.

Usage

```
manual(pkg = ".", path = NULL, preview = TRUE, overwrite = FALSE)
```

Arguments

pkg	package description, can be path or package name. See as.package for more information
path	path in which to produce package. If NULL, defaults to the root directory of the package.
preview	preview generated PDF file
overwrite	overwrite output file if it exists

Author(s)

Mathieu Basille <basille@ase-research.org>

---

mv *Rename an R object.*

Description

Rename an R object.

Usage

```
mv(from, to)
```

Arguments

from	The name of an R object, with or without quotes.
to	The new name, with or without quotes.

Author(s)

Jean Lobry

Examples

```
bla <- 2
ls()
mv(bla, bli)
bli
ls()
```

nselect *Subsetting tables given occurrences*

Description

Select a subset of a table with at least n occurrences of a category.

Usage

```
nselect(x, col, n, droplevels = FALSE)
```

Arguments

x	A data frame or a matrix to be subsetted.
col	The column on which the occurrences are counted; can be the name or the number of the column.
n	The minimum number of occurrences for which to keep the data.
droplevels	Logical. If yes, unused levels from factors in the data frame are dropped.

Value

A data frame.

Author(s)

Mathieu Basille <basille@ase-research.org>

Examples

```
set.seed(1)
bla <- data.frame(value = rnorm(100), group = sample(letters[1:4],
    size = 100, replace = TRUE, prob = (1:4) * 10))
table(bla$group)
bli <- nselect(bla, 2, 25, droplevels = TRUE)
table(bli$group)
```

q *Terminate an R Session*

Description

A modified version of `quit` or its alias `q`. See `quit` for the function details.

Usage

```
q(save = "default", status = 0, runLast = TRUE)

quit(save = "default", status = 0, runLast = TRUE)
```

Details

If `save = "yes"`, the list of attached packages is automatically saved in a file `.Rpackages`. See `savepkglist` for more details.

Author(s)

R Core Team, modified by Mathieu Basille <basille@ase-research.org>

save.image *Save the current workspace*

Description

A modified version of `save.image` that allows to save the commands history and the list of attached packages. See `save.image` for the function details.

Usage

```
save.image(file = ".RData", version = NULL, ascii = FALSE,
  compress = !ascii, safe = TRUE, hist = TRUE, h.file = ".Rhistory",
  pkglist = TRUE, p.file = ".Rpackages")
```

Arguments

`hist`	Logical. Whether to save or not the commands history.
`h.file`	The name of the file in which to save the history, or from which to load it. The path is relative to the current working directory.
`pkglist`	Logical. Whether to save or not the list of attached packages (default is `TRUE`).
`p.file`	The name of the file in which to save the list of attached packages, or from which to load it. The path is relative to the current working directory.

Author(s)

R Core Team, modified by Mathieu Basille <basille@ase-research.org>

See Also

savehistory to save the commands history, and savepkglist to save the list of attached packages.

---

savepkglist *Load or save the list of attached packages*

Description

Load or save the list of attached packages.

Usage

```
savepkglist(file = ".Rpackages")

loadpkglist(file = ".Rpackages")
```

Arguments

file	The name of the file in which to save the list of attached packages, or from which to load it. The path is relative to the current working directory.

Details

savepkglist saves the list of all attached packages, except base packages, in a file, with one package per line.

loadpkglist loads a list of packages from a file. The file should contain one package name per line, without quotes, and no empty line. If the packages are not installed, the function sends a warning.

Note

To automatically load a .Rpackages list at the start of R, add this in your .Rprofile:

```
### Silently load 'basr' together with default packages
options(defaultPackages = c(getOption("defaultPackages"), "basr"))
### Load packages at the start of R if the package list exists
if (file.exists(".Rpackages"))
 basr::loadpkglist(".Rpackages")
```

See Startup for more details about the initialization at start of an R session.

Author(s)

Mathieu Basille <basille@ase-research.org>

Examples

```
## Not run: savepkglist(file = "list.Rpackages")
## Not run: loadpkglist()
```

---

se *Standard errors*

Description

This function computes the standard error (i.e. `sd / sqrt(n)`) of the values in `x`. If `na.rm` is `TRUE` then missing values are removed before computation proceeds.

Usage

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

Arguments

`x`	A numeric vector or an R object which is coercible to one by `as.vector`.
`na.rm`	Logical. Should missing values be removed?

Original URL

http://cran.r-project.org/doc/manuals/R-intro.html

Author(s)

From the `Writing R Extensions` manual, modified by Mathieu Basille <basille@ase-research.org>

See Also

`var` and `sd` for the variance and standard deviation.

Examples

```
bla <- rnorm(1000, sd = 100)
sd(bla)
sqrt(var(bla)/length(bla))
se(bla)

is.na(bla) <- 200:300
sd(bla, na.rm = TRUE)
se(bla, na.rm = TRUE)
```

table *Modified table function to handle NAs*

Description

A slight modification of the table function, to include NA values in the table by default. See table for details of the function.

Usage

```
table(..., exclude = if (useNA == "no") c(NA, NaN), useNA = c("ifany", "no",
  "always"), dnn = list.names(...), deparse.level = 1)
```

Arguments

useNA	Whether to include NA values in the table. Default is now ifany.

Author(s)

R Core Team, modified by Mathieu Basille <basille@ase-research.org>

Examples

```
d <- factor(rep(c("A", "B", "C"), 10), levels = c("A", "B", "C",
    "D", "E"))
is.na(d) <- 3:4
d
table(d)
```

togray *Convert continuous variable to grey levels*

Description

Convert a continuous variable to the corresponding levels of grey.

Usage

```
togray(x, min = 0.1, max = 0.9, alpha = NULL, inverse = FALSE,
  sqrt = FALSE)

togrey(x, min = 0.1, max = 0.9, alpha = NULL, inverse = FALSE,
  sqrt = FALSE)
```

Arguments

x	A numeric vector.
min	The minimum grey level.
max	The maximum grey level.
alpha	The opacity.
inverse	Logical. By default, bigger is darker. If `inverse = TRUE`, bigger is lighter.
sqrt	Logical. Applies a square root transformation to get more progressive grey levels.

Value

A vector of colors of the same length as `x`.

Author(s)

From Clement Calenge, modified by Mathieu Basille <basille@ase-research.org>

Examples

```
bla <- runif(10000)
plot(bla, col = togray(bla, 0, 1), pch = 20)
plot(bla, col = togray(bla, 0, 1, sqrt = TRUE), pch = 20)
plot(bla, col = togray(bla, 0, 1, alpha = 0.5), pch = 20)
```

---

values *Change the values of a vector.*

Description

Replaces given values of a vector by new values.

Usage

```
values(x, from, to)
```

Arguments

x	A character or numeric vector.
from	A vector describing the values to change from.
to	A vector describing the values to change to.

Value

A vector.

Author(s)

Mathieu Basille <basille@ase-research.org>

Examples

```
(bla <- rep(1:5, 3))
values(bla, c(3, 4), c(7, 3))
values(bla, c(3, 4), c("a", "b"))
(bli <- rep(letters[1:5], 3))
values(bli, c("b", "d"), c(1, 2))
blu <- rpois(1e6, 10)
system.time(values(blu, c(3, 4), c(7, 3)))
```

---

writeFunction *Function output*

Description

Prints a function to a file.

Usage

```
writeFunction(fun, file = NULL)
```

Arguments

fun	A function.
file	A character string naming a file. By default, write the function in <fun>.R in the working directory.

Author(s)

Mathieu Basille <basille@ase-research.org>

Examples

```
f1 <- function(x) {
    ## Comment
    print(x)
}
writeFunction(f1)
rm(f1)
source("f1.R")
file.remove("f1.R")
f1(3)
```

Index