Package 'jvamisc'

November 20, 2014

Version 2014-07

Date 2014-07

Title A Collection of Miscellaneous Functions

Author Jean V. Adams [aut, cre]

Maintainer Jean V. Adams <buai.biz@gmail.com>

Depends R (>= 3.1.0)

Imports rtf, MASS, plotrix, lubridate, seriation, survey, devtools

Description jvamisc is a collection of miscellaneous functions.

LazyData TRUE

License GPL

URL https://github.com/JVAdams/jvamisc

R topics documented:

`addedvar` *Added Variable Plots of Predictors*

Description

Produces an added variable plot given 1 response and 2 or more predictors.

Usage

```
addedvar(Y, X, main = "")
```

Arguments

Y	A vector representing a single response.
X	Two or more predictor columns in a matrix or data frame.
main	Subtitle for the plot.

Value

A plot is sent to the current graphics device (no value is returned).

Examples

```
addedvar(Y=mtcars$hp, X=mtcars[, c("mpg", "disp", "wt")],
main="Predicting horsepower from MPG, displacement, and weight")
```

---

addhist *Add a Histogram*

Description

Add a marginal histogram to a plot.

Usage

```
addhist(x, y = NULL, type = "xy", nclass = 20, newmar = 0:1,
  adj.fac = 1.05, xlab = "Frequency", ylab = "", fill = "gray")
```

Arguments

x	A numeric vector, the first set of data to be binned into a histogram.
y	A numeric vector, the second set of data to be binned into a histogram, default NULL.
type	A character scalar, indicating to which plot axes histograms should be added; "x" adds a histogram along the x-axis, "y" adds a histogram along the y-axis, "xy" (the default) adds a histogram along both axes.
nclass	An integer scalar, the target number of bins for the histogram, default 20.
newmar	A numeric vector of length 2, indicating new margins to use, default c(0, 1).
adj.fac	A numeric scalar, adjustment factor for extent of y-axis, default 1.05 (to ensure tallest bars in histogram are below axis box).
xlab	A character scalar, label for x-axis, default "Frequency".
ylab	A character scalar, label for y-axis, default "".
fill	A character or numeric scalar, color for filling in histogram bars, default "gray".

See Also

hist.

Examples

```
# fake data
xx <- rnorm(30)
yy <- runif(30)

# type "x"
layout(matrix(2:1, ncol=1), heights=c(1/5, 4/5))
par(mar=c(4, 4, 0, 1), cex=1, las=1)
plot(xx, yy)
addhist(xx, type="x")

# type "y"
layout(matrix(1:2, ncol=2), widths=c(4/5, 1/5))
par(mar=c(4, 4, 1, 0), cex=1, las=1)
plot(xx, yy)
addhist(yy, type="y")

# type "xy"
layout(matrix(c(2, 1, 0, 3), ncol=2), heights=c(1/5, 4/5), widths=c(4/5, 1/5))
par(mar=c(4, 4, 0, 0), cex=1, las=1)
plot(xx, yy)
usr <- par("usr")
mar <- par("mar")
addhist(xx, yy, type="xy")
```

AICc *Comparison of Models using Akaike Information Criterion*

Description

Compares a collection of statistical models using AIC.

Usage

```
AICc(fitlist, corr = TRUE)
```

Arguments

`fitlist`	A list of model fits to compare, e.g., lm, glm, aov objects.
`corr`	A logical indicating whether the AIC should be corrected for small sample size, default TRUE.

Value

Data frame with a row for each model being compared, ordered by either the uncorrected AIC (corr=FALSE) or the AIC corrected for small sample size (corr=TRUE). Columns include the number of observations (n), the number of parameters (p), the root mean squared error (rmse), the uncorrected AIC (aic), the AIC corrected for small sample size (aicc), the delta AIC (daic or daicc), and the weights of evidence (aicw or aiccw).

Examples

```
fit1 <- lm(hp ~ mpg + disp + wt, data=mtcars)
fit2 <- lm(hp ~ mpg + disp, data=mtcars)
fit3 <- lm(hp ~ mpg + wt, data=mtcars)
fit4 <- lm(hp ~ disp + wt, data=mtcars)
AICc(list(fit1, fit2, fit3, fit4))
```

---

allcombs *All Combinations*

Description

All possible combinations of the given number of items.

Usage

```
allcombs(num, from = 0, to = num)
```

Arguments

num	A scalar, the number of items.
from	A scalar, the minimum number of items in each combination, default 0.
to	A scalar, the maximum number of items in each combination, default num.

Value

A matrix with rows corresponding to each possible combination and columns corresponding to item number.

Examples

```
allcombs(3)
```

---

binomCI *Binomial Confidence Interval*

Description

Calculates the binomial confidence interval from a sample, using the normal approximation. Uses Louis (1981) if only failures or only successes were observed.

Usage

```
binomCI(x, y = NULL, na.rm = TRUE, alpha = 0.05, prob = TRUE)
```

Arguments

x	A vector of 0s and 1s, or (if y is also given) a scalar, the number of failures.
y	A scalar, the number of successes, default NULL.
na.rm	A logical, whether to remove NAs from the data, default TRUE.
alpha	A scalar, the desired confidence level, default 0.05.
prob	A logical, whether to output results as probabilities, default TRUE, or counts.

Value

A named vector with the Mean, lower and upper confidence limits (L and U), and the number of observations N.

References

Thomas A. Louis. 1981. Confidence intervals for a binomial parameter after observing no successes. The American Statistician 35(3):154. *http://amstat.tandfonline.com/doi/abs/10.1080/00031305.1981.10479337?*

Examples

```
binomCI(c(0, 0, 0, 0, 1, 1))
binomCI(4, 2, prob=FALSE)
```

---

blindcolz *Color-blind Friendly Colors*

Description

A vector with nine color-blind friendly colors in hex code.

Usage

```
blindcolz
```

Format

A character vector, length 9.

Author(s)

Masataka Okabe and Kei Ito.

Source

How to make figures and presentations that are friendly to color blind people, 20 November 2002, [link][1].

[1] http://jfly.iam.u-tokyo.ac.jp/html/color_blind/

calcr2 *Coefficient of Determination*

Description

Calculate the unadjusted and adjusted coefficient of determination (R^2).

Usage

```
calcr2(fitted, observed, nparam)
```

Arguments

fitted	A vector of fitted values.
observed	A vector of observed values.
nparam	A scalar, number of parameters in the fitted model.

Value

A vector of unadjusted and adjusted coefficients of determination.

Examples

```
fit1 <- lm(mpg ~ cyl + disp, data=mtcars)
calcr2(fit1$fitted, mtcars$mpg, 3)

fit2 <- lm(mpg ~ cyl + disp + wt, data=mtcars)
calcr2(fit2$fitted, mtcars$mpg, 4)
```

capwords *Capitalize*

Description

Capitalize the first letter of every word.

Usage

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

Arguments

s	A vector of strings.
strict	A logical indicating whether other letters should be converted to lower case, default TRUE.

Value

A vector the same length as s.

See Also

casefold, from which the function was derived.

Examples

```
capwords(c("using AIC for model selection"))
capwords(c("using AIC", "for MODEL selection"), strict=FALSE)
```

---

cheat *Cheat Sheet*

Description

A non-functioning function, which allows me to create a cheat sheet collection of examples.

Usage

```
cheat()
```

Examples

```
## Not run:

# read in xls files
library(XLConnect)
wb <- loadWorkbook("c:/temp/junk.xlsx")
dat <- readWorksheet(wb, sheet=getSheets(wb)[1], startRow=1)

### Greek and math symbols ###
# http://www.decodeunicode.org/
plot(1, 1, xlab="Length  (\U03BCm)", ylab="Temperature  (\U00b0 C)",
main="Lambda squared = \\U03BB\\U00B2 = \U03BB\U00B2")

### update package ###
pkgup("jvamisc")
pkgin("jvamisc")
"C:\Program Files\R\R-3.1.1\bin\x64\R.exe" CMD build C:\JVA\GitHub\jvamisc --resave-data
"C:\Program Files\R\R-3.1.1\bin\x64\R.exe" CMD check C:\Users\jvadams\jvamisc_2014-07.tar
pkgman("jvamisc")

### error bars ###
x <- 1:10
y <- sample(10)
noise <- abs(rnorm(10))
plot(x, y, ylim=range(y-noise, y+noise))
arrows(x, lo, x, hi, length=0.1, angle=90, code=3)

### error shading ###
x <- 1:10
y <- sample(10)
noise <- abs(rnorm(10))
plot(x, y, ylim=range(y-noise, y+noise), type="n")
shadepoly(x, y, y-noise, y+noise)
```

```
### confidence limits ###
# of the mean
p <- predict(fit, interval="confidence")
# of a new observation
p <- predict(fit, interval="prediction")
# if you have a gam ...
p <- predict(fit, se.fit=TRUE)
pe <- p$fit
tt <- qt(1 - 0.05/2, fit$df.residual)
pl <- pe + tt*p$se.fit
pu <- pe - tt*p$se.fit

### multiple comparison Tukey test approach 1 ###
amod <- aov(breaks ~ tension, data = warpbreaks)
TukeyHSD(amod)

### multiple comparison Tukey test approach 2 ###
library(multcomp)
amod <- aov(breaks ~ tension, data = warpbreaks)
mc <- glht(amod, linfct = mcp(tension = "Tukey"))
summary(amod)
summary(mc)
confint(mc)
dev.new()
plot(mc)

### set up two-way ANOVA with interactions ###
fit <- aov(y ~ f1 + f2 + f1:f2)
# set up linear hypotheses for all-pairs of both factors
wht <- glht(fit, linfct = mcp(f1 = "Tukey", f2 = "Tukey"))
# cf. Westfall et al. (1999, page 181)
summary(wht, test = adjusted("Shaffer"))

### label months on day of year or Julian day axis ###
plot(101:200, rnorm(100), axes=FALSE)
axis(1, at=doy(as.Date(paste(2000, 1:12, 1, sep="-")))-0.5,
labels=FALSE)
axis(1, at=doy(as.Date(paste(2000, 1:12, 15, sep="-"))),
labels=month.abb, tick=FALSE)
axis(2)
box()

### contour plot ###
library(akima)
y <- rnorm(50)
x <- runif(50)
z <- 2*x^2 - y^2 + 4
contour(interp(x, y, z, duplicate="mean"))

### get lat longs for locations ###
library(dismo)
geocode(c("1600 Pennsylvania Ave NW, Washington DC",
"Luca, Italy", "Kampala", "Antigo, WI"))

### get a lat long for a location ###
# from R-help post by Phil Spector, UC Berkeley, Mar 16, 2010
# https://stat.ethz.ch/pipermail/r-help/2010-March/232090.html
```

```
library(XML)
root <- xmlRoot(xmlTreeParse(
paste0("http://maps.google.com/maps/api/geocode/xml?address=",
"Antigo, WI", "&sensor=false")))
lat <- xmlValue(root[["result"]][["geometry"]][["location"]][["lat"]])
long <- xmlValue(root[["result"]][["geometry"]][["location"]][["lng"]])

### plot points on a map ###
library(RgoogleMaps)
MyMap <- GetMap.bbox(c(-80, -79), c(45, 46), maptype="terrain",
destfile="junk.png", zoom=8)
PlotOnStaticMap(MyMap, lat=seq(45, 46, 0.1), lon=seq(-80, -79, 0.1),
col="red", pch=16)

### make a quick map ###
library(ggmap)
qmap("Antigo, Wisconsin", zoom=14)

### convert between lat/long and projections ###
library(proj4)
project(xy, proj, inverse=FALSE, degrees=TRUE, silent=FALSE, ellps.default="sphere")

### other map stuff of interest ###
# http://cartodb.com/
# https://github.com/Vizzuality/cartodb-r

### see the details of a function
methods(function)
package:::function.default

### dendrogram reorder ###
x <- sample(100, 10)
names(x) <- x
hc <- hclust(dist(x))
dd <- as.dendrogram(hc)
dd.reorder <- reorder(dd, x, mean)
par(mfcol = 1:2)
plot(dd, main="default dendrogram")
plot(dd.reorder, main="reordered")

### background jpeg image in plot ###
library(jpeg)
x <- rnorm(20)
y <- rnorm(20)
img <- readJPEG("C:/Users/Public/Pictures/Sample Pictures/Chrysanthemum.jpg")
plot(x, y, type="n")
pusr <- par("usr")
rasterImage(img, pusr[1], pusr[3], pusr[2], pusr[4])
points(x, y, pch=16, cex=3)

### fit all subsets models ###
# select all possible combinations of 8 independent variables
var.names <- names(train)[1:8]
comb <- as.data.frame(all.combs(8))
dimnames(comb)[[2]] <- var.names
fits <- vector("list", dim(comb)[1])
fits[[1]] <- lm(lpsa ~ 1, dat=train)
```

```
for(i in 2:length(fits)) {
fits[[i]] <- lm(formula=paste("lpsa ~",
paste(var.names[comb[i, ]==1], collapse=" + ")), dat=train)
}
comb2 <- comb
comb2$nx <- apply(comb, 1, sum)
# AIC
aic <- AICc(fits)
aic <- aic[order(as.numeric(row.names(aic))), -1]
comb2 <- data.frame(comb2, aic)
comb2 <- comb2[order(comb2$aicc), ]
comb2[comb2$daicc <= 2, ]
fits[[as.numeric(row.names(comb2)[1])]]

### regular expression examples ###
# get rid of spaces before commas or periods
t2 <- gsub("[[:space:]]\\.", "\\.", charvec)
gsub("[[:space:]]\\,", "\\,", t2)
# insert a space between all punctuation and letters
gsub("([[:punct:]])([[:alpha:]])", "\\1 \\2", charvec)
# add a period to any single alpha character
t2 <- gsub("([[:space:]][[:alpha:]])([[:space:]])", "\\1\\.\\2", charvec)
gsub("([[:space:]].)$", "\\1\\.", t2)
# insert a space before and after an equal sign
t2 <- gsub("([[:alpha:]]|[[:punct:]])=", "\\1 =", charvec)
gsub("=([[:alpha:]]|[[:punct:]])", "= \\1", t2)
# make sure Jr has a period after it
gsub("Jr$", "Jr.", t2)
# remove all apostrophes (and any surrounding spaces)
t2 <- gsub('[[:space:]]\\"[[:space:]]', "", charvec)
t2 <- gsub('[[:space:]]\\"', "", t2)
gsub('\\"[[:space:]]', "", t2)
# cut off equal sign and everything after
gsub("=.*", "", charvec)
# replace all punctuation marks with spaces
gsub("[[:punct:]]", " ", charvec)
# get rid of leading and trailing white space
gsub("^[ \t]+|[ \t]+$", "", charvec)
# change double spaces to single spaces
gsub("[ \t]+", " ", charvec)

### convert a data frame to json ###
library(RJSONIO)
data <- toJSON(y)
cat(data, file="data.json")

### read in internet table ###
library(XML)
allTables <- readHTMLTable(
"http://en.wikipedia.org/wiki/United_States_presidential_election,_2012")
# Look at the allTables object to find the specific table we want
str(allTables)
# if you have problems reading the URL, you could try this ...
mylines <- readLines(url(
"http://en.wikipedia.org/wiki/United_States_presidential_election,_2012"))
closeAllConnections()
mylist <- readHTMLTable(mylines, asText=TRUE)
```

```
mytable <- mylist1$xTable

### allow users to browse to a file ###
library(tcltk)
myfile <- tk_choose.files(default="C:/JVA/*.csv")

### one slider ###
library(rpanel)
density.draw <- function(panel) {
plot(density(panel$x, bw = panel$h))
panel
}
panel <- rp.control(x = rnorm(50))
rp.slider(panel, h, 0.5, 5, log = TRUE, action = density.draw)

### two sliders ###
library(rpanel)
loess.draw <- function(panel) {
plot(panel$x, panel$y)
lines(loess.smooth(panel$x, panel$y, span=panel$s, degree=panel$d))
panel
}
panel <- rp.control(x=rnorm(50), y=rnorm(50), s=rep(3, 50), d=1)
rp.slider(panel, s, 0.1, 10, showvalue=TRUE, action=loess.draw)
rp.slider(panel, d, 1, 2, showvalue=TRUE, action=loess.draw)

### animation ###
for(i in 1:10) {
dev.new()
plot(1:10, 1:10, type="l")
points(i, i, pch=16, cex=2)
savePlot(filename=paste("Rplot", i), type="bmp")
}
cat(paste("Plots saved to", getwd()), "\n")
# GIF Construction Set
# Animation wizard
# add the bitmaps and save as gif animation file
# GIMP file open as layers ... save as animated gif

## End(Not run)
```

`chi` *Chi-Squared Test*

Description

Performs chi-squared contingency table tests with informative output.

Usage

```
chi(x, rpct = 0, print = TRUE, plot = TRUE)
```

Arguments

`x`	A numeric matrix.
`rpct`	A numeric scalar indicating the rounding used for printed output, default 0.
`print`	A logical scalar indicating if output should be printed, default TRUE.
`plot`	A logical scalar indicating if plot should be generated, default TRUE.

Value

A list with class "htest" containing the components described in `chisq.test`.

See Also

`chisq.test`.

Examples

```
## From Agresti(2007) p.39
M <- as.table(rbind(c(762, 327, 468), c(484, 239, 477)))
dimnames(M) <- list(gender = c("M", "F"),
party = c("Democrat", "Independent", "Republican"))
chi(M)
```

`CI` *Confidence Interval of Mean*

Description

Calculate the confidence interval or limits of the mean using the t distribution.

Usage

```
CI(x, alpha = 0.05, keep.mean = TRUE, limits = TRUE, prefix = "",
  na.rm = FALSE)
```

Arguments

`x`	Numeric vector of sample observations.
`alpha`	Numeric scalar denoting significance level, default 0.05.
`keep.mean`	Logical scalar indicating if the mean should be returned with the confidence interval/limits, default TRUE.
`limits`	Logical scalar indicating if the limits should be returned (otherwise a single interval is returned), default TRUE.
`prefix`	Character scalar to be used in assigning names to the returned value, default "".
`na.rm`	Logical scalar indicating if missing values should be removed before calculations, default FALSE.

Value

A named vector of the mean (if `keep.mean=TRUE`) and the (1 - `alpha`)*100 Names are the `prefix` concatenated to "mean", "lo", and "hi".

Examples

```
CI(1:10)
```

`circles` *Draw Circles on a Plot*

Description

Draw circles on a plot, with control over circle size.

Usage

```
circles(x, y, z, data.range = range(z, na.rm = TRUE),
  circle.size.range = c(0.1, 1), outx = NA, outy = NA, add = FALSE,
  xlim = NULL, ylim = NULL, ...)
```

Arguments

`x`	Numeric vector of x coordinates.
`y`	Numeric vector of y coordinates.
`z`	Numeric vector of data used to generate circles.
`data.range`	Numeric vector, length 2, minimum and maximum `zz` data to plot, default `range(z, na.rm=TRUE)`.
`circle.size.range`	Numeric vector, length 2, minimum and maximum circle radii in inches, default 0.1 to 1.
`outx`	Numeric scalar of x coordinate beyond the figure margins, default NA (see details).
`outy`	Numeric scalar of y coordinate beyond the figure margins, default NA (see details).
`add`	Logical scalar specifying if circles are added to existing plot (TRUE), default FALSE (a new plot is created).
`xlim`	Numeric vector, length 2, x-axis limits, unused if `add=TRUE`.
`ylim`	Numeric vector, length 2, y-axis limits, unused if `add=TRUE`.
`...`	Additional parameters supplied to the `symbols` function.

Details

The size of the circles plotted corresponds directly with the range of data. For example, if there is a `z` of size `data.range[1]`, it will be plotted as a circle with radius `circle.size.range[1]`, and if there is a `zz` of size `data.range[2]`, it will be plotted as a circle with radius `circle.size.range[2]`.

The default of NA for `outx` and `outy` places unseen smallest and biggest circles at a location where the x and y coordinates are 10 times the range observed plus the maximum observed. In most instances this should be well beyond the figure margins.

Value

A data frame with the name, class, dimension, and size of each member of the environment.

Examples

```
circles(trees$Height, trees$Girth, sqrt(trees$Volume),
data.range=sqrt(c(0, max(trees$Volume))), circle.size.range=c(0, 0.3),
xlab="Height (ft)", ylab="Diameter (in)", main="Tree Volume")
```

---

cleanup *Quick Clean Up*

Description

Removes all objects from the current working directory.

Usage

```
cleanup()
```

Details

User is prompted for response. If the first letter of the response is "y" or "Y", all objects are removed from the current working directory.

---

colr *Create a Range of Colors*

Description

Create a range of colors between two specified colors.

Usage

```
colr(x, fromcolname, tocolname)
```

Arguments

x	A numeric vector, the values to be assigned colors.
fromcolname	A character or numeric scalar, indicating the color to use for the lowest value in x. Either a color name (as listed by colors(), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
tocolname	A character or numeric scalar, indicating the color to use for the highest value in x. See fromcolname.

See Also

colors, palette.

Examples

```
x <- 1:10
plot(x, x, pch=16, col=colr(x, "blue", "yellow"), cex=4)
```

coordflip *Flip Coordinates*

Description

Flip (or reflect) coordinates across any given line.

Usage

```
coordflip(pts, seg)
```

Arguments

pts	A numeric matrix with two columns of x and y coordinates to be rotated.
seg	A numeric matrix of dimension 2 x 2 giving two points which define the line over which the coordinates will be flipped.

Value

A numeric matrix with same dimension as `pts` with the flipped x and y coordinates.

References

Based on a method posted by Il-Bhima on 22 July 2010 on stackoverflow [link][2].

See Also

`coordplot`, `coordmove`, `coordturn`, `coordtri`.

Examples

```
# starting coordinates
test <- matrix(c(0, 4, 1, 0, 2, 3), ncol=2, dimnames=list(LETTERS[1:3], NULL))
coordplot(test)
# flip the coordinates across the line defined by the first two points
ftest <- coordflip(test, test[1:2, ])
coordplot(ftest)
```

coordmove *Move Coordinates*

Description

Move coordinates in the x and y direction in the plane.

Usage

```
coordmove(pts, from, to)
```

[2] http://stackoverflow.com/questions/3306838/algorithm-for-reflecting-a-point-across-a-line

Arguments

pts	A numeric matrix with two columns of x and y coordinates to be moved.
from	A numeric vector of length 2, the starting x and y coordinates of a point.
to	A numeric vector of length 2, the ending x and y coordinates of a point.

Value

A numeric matrix with same dimension as pts with the moved x and y coordinates.

See Also

coordplot, coordturn, coordflip, coordtri.

Examples

```
test <- matrix(c(0, 4, 1, 0, 2, 3), ncol=2, dimnames=list(LETTERS[1:3], NULL))
coordplot(test)
# move the coordinates so that the second point is at the origin
mtest <- coordmove(test, test[2, ], c(0, 0))
coordplot(mtest)
```

---

coordplot *Plot Coordinates*

Description

Quick plot of coordinates on equally scaled coordinate plane, labelled with row names.

Usage

```
coordplot(pts)
```

Arguments

pts	A numeric matrix with two columns of x and y coordinates to be plotted.

See Also

coordturn, coordmove, coordflip, coordtri.

Examples

```
test <- matrix(c(0, 4, 1, 0, 2, 3), ncol=2, dimnames=list(LETTERS[1:3], NULL))
coordplot(test)
```

coordtri *Determine Third Coordinate of Triangle*

Description

Determine the third coordinate of a triangle, given the other two coordinates and the distances to the third.

Usage

```
coordtri(cleft, cright, lleft, lright)
```

Arguments

cleft	A numeric vector of length 2, the x and y coordinates of the lower left point of the triangle (see details).
cright	A numeric vector of length 2, the x and y coordinates of the lower right point of the triangle (see details).
lleft	A numeric scalar, the length of the left side of the triangle, from cleft to the top (see details).
lright	A numeric scalar, the length of the right side of the triangle, from cright to the top (see details).

Details

The directions "left" and "right" refer to the triangle not necessarily in its current state, but rather **after** it has been rotated such that the bottom of the triangle is level (e.g., on the y=0 line), and the third coordinate that the function returns is "above" the bottom (e.g., y>0).

A warning message is given if a triangle can't be built from the inputs provided, and the returned value is c(NA, NA).

Value

A numeric vector of length 2, the x and y coordinates of the top point of the triangle.

See Also

coordplot, coordturn, coordmove, coordflip.

Examples

```
# define coordinates of base of triangle
AB <- rbind(A=c(-2, 4), B=c(2, -4))
# determine coordinates of top of triangle
# given base coordinates and side lengths
C <- coordtri(cleft=AB[1, ], cright=AB[2, ], lleft=12, lright=10)
# plot results
coordplot(rbind(C, AB))
```

coordturn *Rotate Coordinates*

Description

Rotate coordinates clockwise around a pivot point.

Usage

```
coordturn(pts, pvt, rot)
```

Arguments

pts	A numeric matrix with two columns of x and y coordinates to be rotated.
pvt	A numeric vector of length 2, the x and y coordinates of the pivot point around which pts will be rotated.
rot	A numeric scalar indicating the amount of clockwise rotation, in radians.

Value

A numeric matrix with same dimension as pts with the rotated x and y coordinates.

References

Modification of code posted by Sage on 3 March 2011 on the website benn.org [link][3].

See Also

coordplot, coordmove, coordflip, coordtri.

Examples

```
# starting coordinates
test <- matrix(c(0, 4, 1, 0, 2, 3), ncol=2,
dimnames=list(LETTERS[1:3], NULL))
coordplot(test)
# rotate the coordinates clockwise 45 degrees
# around the first point (the origin)
rottest <- coordturn(test, test[1, ], rot=pi/4)
coordplot(rottest)
# rotate the coordinates counterclockwise 45 degrees
# around the first point (the origin)
rottest <- coordturn(test, test[1, ], rot=-pi/4)
coordplot(rottest)
```

[3] http://benn.org/2007/01/06/rotating-coordinates-around-a-centre/

dfclip *Read Data Frame from Clipboard*

Description

Read in a data frame from text pasted to the clipboard.

Usage

```
dfclip(...)
```

Arguments

`...`	Additional parameters to `read.table`.

Value

Data frame.

See Also

`read.table`.

doy *Day of Year*

Description

Calculates the day of the year.

Usage

```
doy(date, day1 = "01-01")
```

Arguments

`date`	A vector of dates to be converted.
`day1`	A character scalar specifying what date should equate to day 1, use "01-01", the default, to get the Julian day.

Value

A numeric vector the same length as `date` giving the day of the year.

Examples

```
x <- as.Date(c("1963-01-15", "1972-07-20", "1999-03-10"))
doy(x)
doy(x, "03-01")
```

endrtf *Write and Close an RTF Document*

Description

Write and close an rtf (rich text format) document.

Usage

```
endrtf(rtf = doc, details = FALSE, ...)
```

Arguments

rtf	An rtf object, default doc.
details	Logical scalar indicating if session details should be added to the end of the document, default FALSE.
...	Additional parameters to addPageBreak.

See Also

startrtf, heading, para, tabl, figu, figbig, RTF, addPageBreak.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

figbig *Add a Big Figure to an RTF Document*

Description

Add a big figure to an rtf (rich text format) document.

Usage

```
figbig(..., FIG = fig, rtf = doc, figc = figcount, boldt = TRUE,
  w = NULL, h = NULL, rf = 300, newpage = "none", omi = c(1, 1, 1, 1))
```

Arguments

...	One or more character scalars (separated by commas) of text to use for the figure caption.
FIG	A function to create a figure which will be added to the document, default fig.
rtf	An rtf object, default doc.
figc	Numeric scalar figure number to use in caption, default figcount.
boldt	Logical scalar indicating if figure number should use bold font, default TRUE.
w	Numeric scalar width of figure in inches, default 6.5.
h	Numeric scalar height of figure in inches, default 8.
rf	Numeric scalar resolution of figure, default 300.
newpage	Character scalar indicating if the figure should start on a new page in the document "port" for a new portrait page, "land" for a new landscape page, and "none" for no new page (the default).
omi	Numeric vector, length 4, width of document page margins in inches (bottom, left, top, right), default c(1, 1, 1, 1).

Details

The figure and caption are written to the rtf file. The size of a new page is assumed to be 11 by 17 inches.

Value

A 1 is added to the numeric vector of length 1, figcount, stored in the working directory to keep track of the number of figures written to the rtf document, and label the captions accordingly.

See Also

startrtf, heading, para, tabl, endrtf, RTF.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figbig("Speed vs. distance from the cars data.", newpage="land")
```

```
endrtf()

## End(Not run)
```

---

figu *Add a Figure to an RTF Document*

Description

Add a figure to an rtf (rich text format) document.

Usage

```
figu(..., FIG = fig, rtf = doc, figid = "Figure ", fign = figcount,
  boldt = TRUE, capunder = TRUE, w = NULL, h = NULL, rf = 300,
  newpage = "none", omi = c(1, 1, 1, 1))
```

Arguments

`...`	One or more character scalars (separated by commas) of text to use for the figure caption.
`FIG`	A function to create a figure which will be added to the document, default `fig`.
`rtf`	An rtf object, default `doc`.
`figid`	Character scalar of caption identifier, default "Figure ".
`fign`	Numeric scalar of figure number to use in caption,, default `figcount`.
`boldt`	Logical scalar indicating if figure number should use bold font, default TRUE.
`capunder`	Logical scalar indicating if caption should appear under the figure (TRUE, the default) or on top of the figure (FALSE).
`w`	Numeric scalar width of figure in inches, default 6.5.
`h`	Numeric scalar height of figure in inches, default 8.
`rf`	Numeric scalar resolution of figure, default 300.
`newpage`	Character scalar indicating if the figure should start on a new page in the document "port" for a new portrait page, "land" for a new landscape page, and "none" for no new page (the default).
`omi`	Numeric vector, length 4, width of document page margins in inches (bottom, left, top, right), default c(1, 1, 1, 1).

Details

The figure and caption are written to the rtf file. The size of a new page is assumed to be 8.5 by 11 inches.

Value

A 1 is added to the numeric vector of length 1, `figcount`, stored in the working directory to keep track of the number of figures written to the rtf document, and label the captions accordingly.

See Also

startrtf, heading, para, tabl, figbig, endrtf, RTF.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

---

fill *Fill in Missing Values*

Description

Fill in missing values in a vector, using the last recorded value.

Usage

```
fill(x)
```

Arguments

x	A vector, can be character, numeric, or logical.

Details

Similar to na.locf in the zoo package, but works for "" in character vectors as well.

Value

A vector the same length as x, with all NAs or ""s replace by the last value for the vector. Note that and missing values at the beginning of the vector will not be replaced.

Examples

```
numvec <- c(NA, 1:5, NA, NA, NA, 10:12, NA)
fill(numvec)

charvec <- c("", letters[1:5], "", "", "", letters[10:12], "")
fill(charvec)
```

first *Identify the First Elements of Series of Repeated Values*

Description

Identify the first elements of series of repeated values.

Usage

```
first(x)
```

Arguments

x	A vector whose values will be explored for series of repeated values, can be character, numeric, or factor.

Value

An integer vector the same length as x, with a 1 for every element that is different than the one before it, and a 0 for every element that is the same as the one before it.

Examples

```
first(c(1, 2, 1, 2, 2, 1, 1, 3))
```

formatdf *Format a Data Frame*

Description

Format selected columns of a data frame for pretty viewing or printing.

Usage

```
formatdf(df, numercol = NULL, round2 = 0, ndec = round2, comma = ",",
  characol = NULL, align = "left", keepnum = TRUE)
```

Arguments

df	Data frame to be formatted.
numercol	Numeric vector, index of numeric columns to format, default, NULL, uses all numeric and integer columns.
round2	Integer vector, indicating what place numbers should be rounded to, default 0.
ndec	Integer vector, indicating how many places to the right of the decimal point should be displayed, default round2.
comma	Character vector, indicating thousands character separator, default ",".
characol	Numeric vector, index of character columns to format, default, NULL, uses all character and factor columns.

`align`	Character vector, indicating justification of character columns, default "left". Other options include "right", "centre", and "none".
`keepnum`	Logical scalar, indicating if numeric columns should be kept as numeric, default TRUE, or converted to character.

Details

If a vector of length 1 is provided for `round2`, `ndec`, `comma`, or `align`, the same rounding, decimals, commas, or alignment is applied to all specified columns.

Value

A data frame, the same dimensions as `df` with selected columns rounded and/or formatted and converted to character.

See Also

`format`.

Examples

```
head(mtcars)
formatdf(head(mtcars))
```

`getpackages` *Get Packages*

Description

Installs (if necessary) and attaches the specified packages.

Usage

```
getpackages(want)
```

Arguments

`want`	A character vector of package names.

heading *Add a Heading to an RTF Document*

Description

Add a text heading to an rtf (rich text format) document.

Usage

```
heading(words, htype = 1, rtf = doc)
```

Arguments

words	Character scalar text of heading to add to document.
htype	Integer scalar heading type, 1 = bold and font size 12, 2 = bold and font size 10, 3 = italics and font size 10, default 1.
rtf	An rtf object, default doc.

Details

The specified heading is written to the rtf file.

See Also

startrtf, para, tabl, figu, figbig, endrtf, RTF.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

inrange *In Range*

Description

Test to see if value falls within specified range.

Usage

```
inrange(x, r, open = TRUE)
```

Arguments

x	A numeric vector of values to be tested.
r	A numeric vector of length 2 specifying the range.
open	A logical scalar indicating if the range interval is open (excludes endpoints, TRUE, default) or closed (includes endpoints, FALSE).

Value

A logical vector, the same length as x indicating if values fall within the specified range.

Examples

```
inrange(4:6, c(4, 5.5))
inrange(4:6, c(4, 5.5), FALSE)
```

jvaFirst *Startup*

Description

One function with all the commands I typically want run at the start of an R session.

Usage

```
jvaFirst(maxp = 500, ndec = 10, cont = "... ", pkgs = c("rJava",
  "XLConnect", "maps", "mapproj", "RColorBrewer", "mgcv", "MASS", "jvamisc"),
  mirror = "http://streaming.stat.iastate.edu/CRAN/", helpt = "html",
  fac = FALSE, noplots = TRUE, show = TRUE)
```

Arguments

maxp	Integer scalar, maximum number of lines printed, default 500.
ndec	Integer scalar, maximum width of fixed notation before switching to scientific notation, default 10.
cont	Character scalar, prompt used for lines which continue past first command line, default "... ".
pkgs	Character vector, packages to be loaded, default c("rJava", "XLConnect", "maps", "mapproj", "RColorBrewer", "mgcv", "jvamisc").
mirror	Character scalar, CRAN mirror, default "http://streaming.stat.iastate.edu/CRAN/".
helpt	Character scalar, type of help, default "html".
fac	Logical scalar, use factors rather than character strings, default FALSE.
noplots	Logical scalar, remove "saved" plots, e.g., from past runs of plotdf, default TRUE.
show	Logical scalar, list objects in current environment, default TRUE.

See Also

options, help.

Examples

```
jvaFirst()
```

---

jvaLast *Shutdown*

Description

One function with all the commands I typically want run at the end of an R session.

Usage

```
jvaLast(file = ".Rhistory", nlines = 10000)
```

Arguments

file	Character scalar, file in which to save the commands history relative to current working directory, default ".Rhistory".
nlines	Integer scalar, number of lines that saved to commands history, default 10,000.

See Also

savehistory, Sys.setenv.

jvamisc *A Collection of Miscellaneous Functions*

Description

jvamisc is simply a collection of miscellaneous functions that I find useful. Some of the more commonly used functions include `plotdf` to plot each variable of a data frame, `shadepoly` to add a shaded polygon of confidence intervals to a plot, `AICc` to compare models using Akaike's Information Criterion, and `startrtf`, `heading`, `para`, `tabl`, `figu`, and `endrtf` to write text, tables, and figures to a Word document.

jvanames *Format Names*

Description

Format names to be all lower case, unique, and without underscores.

Usage

```
jvanames(charvec)
```

Arguments

`charvec`	A character vector to be trimmed.

Value

A character vector, the same length as `charvec`, but with all lower case, unique, and without underscores.

See Also

`make.names`

Examples

```
jvanames(c("Conc_mgL", "Conc_mgL", "Temp"))
```

Lakeabbs *Great Lakes Abbreviations*

Description

A vector with the first initials of the five Great Lakes.

Usage

```
Lakeabbs
```

Format

A character vector, length 5.

Lakenames *Great Lakes Names*

Description

A vector with the names of the five Great Lakes.

Usage

```
Lakenames
```

Format

A character vector, length 5.

last *Identify the Last Elements of Series of Repeated Values*

Description

Identify the last elements of series of repeated values.

Usage

```
last(x)
```

Arguments

x	A vector whose values will be explored for series of repeated values, can be character, numeric, or factor.

Value

An integer vector the same length as `x`, with a 1 for every element that is different than the one after it, and a 0 for every element that is the same as the one after it.

Examples

```
last(c(1, 2, 1, 2, 2, 1, 1, 3))
```

`lls` *List All Members of an Environment.*

Description

Generate a descriptive list of all members of an environment.

Usage

```
lls(pos = 1, pat = "")
```

Arguments

`pos`	A scalar of the environment, search path index, or package of interest, default 1.
`pat`	Regular expression pattern specifying which members to return, default "" for all members.

Value

A data frame with the name, class, dimension, and size of each member of the environment.

References

Based on a method posted by Bendix Carstensen on 10 Jan 2007 on R-help [link][4].

Examples

```
## Not run:
lls()

## End(Not run)
```

[4] `https://stat.ethz.ch/pipermail/r-help/2007-January/123403.html`

map5 *Great Lakes Basin Shoreline*

Description

A single data frame with longitude and latitude coordinates of the shorelines in the Great Lakes basin.

Usage

```
map5
```

Format

One row for each point, consecutive rows with non-missing values for each line.

Author(s)

Rich Signell, USGS.

Source

NOAA National Geophysical Data Center, (the now defunct) Coastline Extractor, ngdc.noaa.gov/mgg/coast[5].

mapL *Great Lakes Shorelines*

Description

List of five data frames, one for each Great Lake with longitude and latitude coordinates of the shorelines.

Usage

```
mapL
```

Format

One row for each point, consecutive rows with non-missing values for each line.

Author(s)

Rich Signell, USGS.

Source

NOAA National Geophysical Data Center, (the now defunct) Coastline Extractor, ngdc.noaa.gov/mgg/coast[6].

[5] http://www.ngdc.noaa.gov/mgg/coast/
[6] http://www.ngdc.noaa.gov/mgg/coast/

matrixtrim *Trim Matrix*

Description

Trim the rows and columns of a logical matrix until every row and columns has a specified level of TRUE values.

Usage

```
matrixtrim(m, prop = c(1, 1), rowsfirst = TRUE)
```

Arguments

m	A logical matrix to be trimmed, should be all TRUE, FALSE (or 0, 1).
prop	A numeric vector of length two, the minimum proportion of TRUE values in each row and column of the trimmed matrix, default c(1, 1).
rowsfirst	A logical scalar, indicating if rows (TRUE, the default) or columns (FALSE) should be trimmed from the matrix first.

Value

A list of length three:

- trim = a logical matrix, the resulting trimmed matrix
- dim = a numeric vector of length 2, the dimensions of trim
- n = a numeric scalar, the total number of TRUE values in trim

Examples

```
m <- matrix(rep(c(1, 0, 1, 0), c(4, 1, 13, 2)), nrow=5)
matrixtrim(m)
matrixtrim(m, rowsfirst=FALSE)
matrixtrim(m, prop=c(0.7, 0.7))
```

modecontin *Mode of Continuous Variable*

Description

Estimates the mode of a continuous variable.

Usage

```
modecontin(x, plot = FALSE, ...)
```

Arguments

x	A numeric vector.
plot	A logical scalar indicating whether to plot the results, default FALSE.
...	Additional arguments to the density function.

Value

The estimated mode of x.

References

Based on a method posted by Peter Dalgaard on 29 Aug 2008 on r-help[7].

Examples

```
x <- rnorm(100)
modecontin(x, TRUE)
```

---

para *Add a Paragraph to an RTF Document*

Description

Add a paragraph to an rtf (rich text format) document.

Usage

```
para(..., rtf = doc, bold = FALSE, italic = FALSE)
```

Arguments

...	One or more character scalars (separated by commas) of text to add to document as a single paragraph.
rtf	An rtf object, default doc.
bold	Logical scalar indicating if paragraph should use bold font, default FALSE.
italic	Logical scalar indicating if paragraph should use italic font, default FALSE.

Details

The specified heading is written to the rtf file.

See Also

startrtf, heading, tabl, figu, figbig, endrtf, RTF.

[7] https://stat.ethz.ch/pipermail/r-help/2008-August/172319.html

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

---

pkgin *Install a Package*

Description

Install a package from local files, load the library, and save installed package to zip archive.

Usage

```
pkgin(package, wd = "C:/JVA/R/Working Directory",
  ld = "C:/Users/jvadams/Documents/R/win-library/3.1", pd = "C:/JVA/GitHub")
```

Arguments

package	A character scalar, package name.
wd	A character scalar, R working directory, default "C:/JVA/R/Working Directory".
ld	A character scalar, R library directory, default "C:/Users/jvadams/Documents/R/win-library/3.1".
pd	A character scalar, R package directory, default "C:/JVA/GitHub".

See Also

pkgup

pkgman *Copy a Package Manual*

Description

Copy a package reference manual created by R CMD Check to the corresponding GitHub folder.

Usage

```
pkgman(package, file = paste0("C:/Users/jvadams/", package, ".Rcheck/",
  package, "-manual.pdf"), dir = "C:/JVA/GitHub/")
```

Arguments

package	A character scalar, package name.
file	A character scalar, path of file to be copied, default paste0("C:/Users/jvadams/", package, ".Rcheck/", package, "-manual.pdf").
dir	A character scalar, directory where package folder is located, default "C:/JVA/GitHub/".

See Also

pkgin

pkgup *Document a Package*

Description

Document a package from local files.

Usage

```
pkgup(package, dir = "C:/JVA/GitHub/")
```

Arguments

package	A character scalar, package name.
dir	A character scalar, directory where package folder is located, default "C:/JVA/GitHub/".

See Also

pkgin

plotblank *Create a Blank Plot*

Description

Create a blank plot (no symbols) on which to add other plotting features.

Usage

```
plotblank(x = 0:1, y = 0:1, xlab = "", ylab = "", las = 1, ...)
```

Arguments

x	A numeric vector, the x coordinates of points in the plot.
y	A numeric vector, the y coordinates of points in the plot.
xlab	A character scalar, title for the x axis, default "".
ylab	A character scalar, title for the y axis, default "".
las	A numeric scalar, style of axis labels, 0 = always parallel to the axis, 1 = always horizontal (default), 2 = always perpendicular to the axis, 3 = always vertical.
...	Additional arguments to the plot function.

See Also

plot, title, par

Examples

```
plotblank(xlim=c(1, 100))
```

plotcor *Plot a Matrix of Correlations*

Description

Plot a matrix of correlations using white ellipses (representing the measured correlation) overlaid on colored circles.

Usage

```
plotcor(r, addtext = TRUE, atcex = NULL, incdiag = FALSE, rorder = TRUE,
  ...)
```

Arguments

r	A matrix of correlations (need not be symmetrical) with values ranging from -1 to 1.
addtext	A logical scalar indicating whether the value of each correlation should be written over the ellipses, default TRUE.
atcex	A numeric scalar giving the magnification of the added ellipse text relative to that set in par, default NULL results in atcex = 8/maximum dimension of r.
incdiag	A logical scalar indicating whether to plot circles for the diagonal values of the matrix (if symmetric), default FALSE.
rorder	A logical scalar indicating whether the columns and rows of the matrix should be reordered using seriation, default TRUE.
...	Additional parameters to par.

Details

Each ellipse is sized so that the proportion of the colored circle visible beyond the ellipse is equal to the squared correlation. Each ellipse is oriented northeast for positive correlation and northwest for negative correlation. The color of each circle ranges from cyan (for correlation = -1) to magenta (for correlation = 1) through white (for correlation = 0). Similarly, the transparency of each correlation value (if addtext=TRUE) ranges from 1 (for correlation = 0) to 0 (for absolute correlation = 1).

Value

A list of with two vector of integers (the same length as each dimension of r) representing the linear order suggested by seriation.

See Also

plotcorr on which the idea for the function was based, seriate, and draw.ellipse.

Examples

```
# example using a symmetric matrix
sr <- cor(swiss)
sord <- plotcor(sr)
sr[sord[[1]], sord[[2]]]
# example using an asymmetric matrix
lr <- cor(longley)[1:3, 4:7]
lord <- plotcor(lr)
lr[lord[[1]], lord[[2]]]
```

---

plotdf *Plot Data Frame*

Description

Plots each variable of a data frame.

Usage

```
plotdf(df, one = TRUE, ...)
```

Arguments

df	A data frame.
one	A logical, whether to display all plots in a single graphics device, default TRUE.
...	Other parameters to par(...).

Value

A data frame with summary statistics for each variable in the data frame, number entered, number missing, number unique, minimum, mean, maximum, and the ratio of the maximum over the non-zero minimum.

See Also

par.

Examples

```
plotdf(mtcars)
```

---

prettylog *Pretty Breakpoints on Log Scale*

Description

Compute a sequence of "round" values which cover the range of x on the log scale.

Usage

```
prettylog(x, lead = c(1, 5), extra = 5)
```

Arguments

x	A numeric vector.
lead	An integer vector giving the desired lead digits of pretty values on the log scale, default c(1, 5).
extra	An integer scalar giving the desired number of additional non-log scale values to include, default 5.

Value

A numeric vector of pretty values covering the range of x on the log scale.

Examples

```
vals <- rlnorm(100, 6)
summary(vals)
prettylog(vals, 1, 0)
prettylog(vals, 1)
prettylog(vals, c(1, 2, 5))
```

prettytable *Prettify the Numeric Columns of a Table*

Description

Prettify the numeric columns of a table for printing.

Usage

```
prettytable(m, sigdig = 3)
```

Arguments

m	Two dimensional data frame, matrix, or table to be prettified.
sigdig	Integer scalar, number of significant digits to be used, default 3.

Value

A data frame, matrix, or table, the same dimensions as m with the numeric columns rounded to sigdig significant digits.

See Also

signif.

Examples

```
head(mtcars)
prettytable(head(mtcars), 2)
```

q *Quit*

Description

Terminate the current R session.

Usage

```
q(save = "yes")
```

Arguments

save	A character string indicating whether the workspace should be saved, "yes" (the default), "no", "ask" or "default".

See Also

quit.

`ratest` *Ratio Estimation*

Description

Ratio estimation.

Usage

```
ratest(num, den, adj = 1)
```

Arguments

`num`	A numeric vector, the numerator of the ratio.
`den`	A numeric vector, the denominator of the ratio.
`adj`	A numeric scalar, an adjustment factor to be multiplied by the ratio, default 1.

Details

All records with a missing value in either the numerator or the denominator are omitted from calculations.

Value

A named vector with the sum of the numerators, the sum of the denominators, the ratio of the sums, the number of non-missing input pairs, and the standard deviation and 95

See Also

`svydesign`, `svyratio`.

Examples

```
# Size, weekly income, and food cost of 33 families from
# Cochran's (1977) Sampling Techniques.
familysize <- c(2, 3, 3, 5, 4, 7, 2, 4, 2, 5, 3, 6, 4, 4,
2, 5, 3, 4, 2, 4, 2, 5, 3, 4, 7, 3, 3, 6, 2, 2, 6, 4, 2)
income <- c(62, 62, 87, 65, 58, 92, 88, 79, 83, 62, 63, 62, 60, 75, 90,
75, 69, 83, 85, 73, 66, 58, 77, 69, 65, 77, 69, 95, 77, 69, 69, 67, 63)
foodcost <- c(14.3, 20.8, 22.7, 30.5, 41.2, 28.2, 24.2, 30, 24.2, 44.4,
13.4, 19.8, 29.4, 27.1, 22.2, 37.7, 22.6, 36, 20.6, 27.7, 25.9, 23.5,
39.8, 16.8, 37.8, 34.8, 28.7, 63, 19.5, 21.6, 18.2, 20.1, 20.7)
# weekly expenditure on food per person
ratest(foodcost, familysize)
# percentage of income spent on food
ratest(foodcost, income, 100)
```

recode — *Recode Values*

Description

Assign new values to a vector.

Usage

```
recode(x, old, new, must.match = TRUE)
```

Arguments

x	A vector whose values will be recoded, can be character, numeric, or factor.
old	A vector of the unique values currently in the vector.
new	A vector of values which should replace the current ones.
must.match	A logical scalar indicating whether only those elements of the original vector with values in old should be returned (TRUE), or all values should be returned (FALSE) though some may be unchanged, default TRUE.

Value

A vector the same length as x (unless must.match=TRUE), with old values replaced by new values.

Examples

```
recode(c(1,1,1,2,3,4,1,10,3), 1:3, 1001:1003)
recode(c(1,1,1,2,3,4,1,10,3), 1:3, 1001:1003, must.match=FALSE)
```

segreg — *Segmented Regression*

Description

Fit a simple linear segmented regression (also known as changepoint, hockey stick, or broken line regression).

Usage

```
segreg(x, y, k = NULL)
```

Arguments

x	Numeric vector, the independent variable which will be "broken".
y	Numeric vector, the dependent variable which will be used to define the break point.
k	Numeric scalar, the location of the break point, if known. If NULL, the default, the optimal break point will be chosen from among all unique values of x except for the minimum and the maximum.

Details

The break point cuts the x vector into two groups, x <= k and x > k.

Value

List with three elements, fit: the resulting lm object, k, and pred: data frame with the unique values of x (and k) with predicted values.

Examples

```
indvar <- sample(1:50, 100, replace=TRUE)
depvar <- ifelse(indvar<32, indvar + 3, indvar/4 + 27) + rnorm(100, sd=2)
sr <- segreg(indvar, depvar)
plot(indvar, depvar)
lines(sr$pred)
abline(v=sr$k, lty=2)
```

---

shadepoly *Add Shaded Polygon to Plot*

Description

Add a spline-smoothed, shaded polygon to a plot, typically to show an interval or range of values (in the y-direction) for a time series of x values.

Usage

```
shadepoly(x, ymd, ylo, yhi, subsel = NULL, kol = "#000000", opq = c(20,
  50), addline = TRUE)
```

Arguments

x	A numeric vector, the metric to plot in the x direction (e.g., time).
ymd	A numeric vector, the metric to plot in the y direction (e.g., a mean or median).
ylo	A numeric vector, the lower interval or range in the y direction (e.g., a lower confidence interval or quartile).
yhi	A numeric vector, the upper interval or range in the y direction (e.g., an upper confidence interval or quartile).
subsel	A logical vector, indicating subset of the data to plot.
kol	A character scalar, the hex color to use for plotting both the shaded polygon and (if requested) the line, default "#000000" (black).
opq	A numeric vector of length 2, opacity for the polygon and the line, default c(20, 50).
addline	A logical scalar, indicating if the x vs ymd line should be added to the plot (on top of the shaded polygon), default TRUE.

Details

Missing values are removed prior to plotting, such that there will be no breaks in the shaded polygon nor the line (if requested). The lower and upper intervals of the polygon are spline-smoothed prior to plotting, as is the line (if requested).

See Also

polygon, lines.

Examples

```
x <- 1:10
y <- sample(10)
noise <- abs(rnorm(10))
plot(x, y, ylim=range(y-noise, y+noise), type="n")
shadepoly(x, y, y-noise, y+noise)
```

---

showmarks *Show Marks*

Description

Show marks used in graphing, including line types, plotting symbols, default colors, color blind friendly colors (‘blindcolz‘), and fonts.

Usage

```
showmarks()
```

Examples

```
showmarks()
```

---

startrtf *Create an RTF Document*

Description

Create an rtf (rich text format) document.

Usage

```
startrtf(file = NULL, dir = getwd(), width = 8.5, height = 11,
  omi = c(1, 1, 1, 1), quiet = FALSE)
```

Arguments

file	Character scalar name of document, default "RGeneratedDocument" with Sys.Date suffix.
dir	Character scalar name of directory where document should be stored, default getwd().
width	Numeric scalar width of document page in inches, default 8.5.
height	Numeric scalar height of document page in inches, default 11.
omi	Numeric vector, length 4, width of document page margins in inches (bottom, left, top, right), default c(1, 1, 1, 1).
quiet	Logical scalar indicating if name of new rtf document should be printed to command line, default FALSE.

Details

The rtf file may be written to until the `endrtf()` function is run. If you assign your rtf file to an object called `doc`, you can use the default settings in other **jvamisc** rtf functions.

Value

An rtf file is created in the specified directory. An object of class rtf is created. This object is referred to in other function to write to the file. In addition, two numeric vectors of length 1, `tabcount` and `figcount`, are written to the working directory to keep track of the number of tables and figures written to the rtf document, and label the captions accordingly.

See Also

`heading`, `para`, `tabl`, `figu`, `figbig`, `endrtf`, `RTF`.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

`stringin` *Find a String in a Character Vector*

Description

Find a string in a character vector.

Usage

```
stringin(pattern, x, ignore.case = TRUE, value = TRUE, fixed = TRUE, ...)
```

Arguments

`pattern`	Character scalar, string to be matched in `x`.
`x`	Character vector where matches are sought.
`ignore.case`	Logical scalar, indicating case sensitivity, default TRUE.

value	Logical scalar, indicating whether to return the matching elements (the default, TRUE) or their indices (FALSE).
fixed	Logical scalar, indicating whether string should be matched as is, default TRUE.
...	Other arguments to grep.

Value

Character vector containing the matching elements of x.

See Also

grep.

Examples

```
txt <- c("The", "licenses", "for", "most", "software", "are", "designed",
"to", "take", "away", "your", "freedom", "to", "share", "and", "change",
"it.", "", "By", "contrast,", "the", "GNU", "General", "Public", "License",
"is", "intended", "to", "guarantee", "your", "freedom", "to", "share", "and",
"change", "free", "software", "--", "to", "make", "sure", "the", "software",
"is", "free", "for", "all", "its", "users.")
stringin("b", txt)
stringin("ar", txt)
```

---

tabl *Add a Table to an RTF Document*

Description

Add a table to an rtf (rich text format) document.

Usage

```
tabl(..., TAB = tab, rtf = doc, fontt = 8, row.names = TRUE,
  tabc = tabcount, boldt = TRUE, newpage = "none", omi = c(1, 1, 1, 1))
```

Arguments

...	One or more character scalars (separated by commas) of text to use for the table caption.
TAB	A matrix, data frame, or table to be added to the document as a table, default tab.
rtf	An rtf object, default doc.
fontt	Numeric scalar font size for table caption, default 8.
row.names	Logical scalar whether to include the row.names of TAB in the table, default TRUE.
tabc	Numeric scalar table number to use in caption, default tabcount.
boldt	Logical scalar indicating if table number should use bold font, default TRUE.

newpage	Character scalar indicating if the table should start on a new page in the document "port" for a new portrait page, "land" for a new landscape page, and "none" for no new page (the default).
omi	Numeric vector, length 4, width of document page margins in inches (bottom, left, top, right), default c(1, 1, 1, 1).

Details

The table and caption are written to the rtf file. The size of a new page is assumed to be 8.5 by 11 inches.

Value

A 1 is added to the numeric vector of length 1, tabcount, stored in the working directory to keep track of the number of tables written to the rtf document, and label the captions accordingly.

See Also

startrtf, heading, para, figu, figbig, endrtf, RTF.

Examples

```
## Not run:
doc <- startrtf()
heading("Heading 1")
para("First paragraph.")
tab <- head(cars)
tabl("First few rows of cars data.", row.names=FALSE)
heading("Heading 2", 2)
para("Second paragraph.")
fig <- function() {
plot(cars)
lo <- loess(cars$dist ~ cars$speed)
lines(lo$x, lo$fitted)
}
figu("Speed vs. distance from the cars data.")
endrtf()

## End(Not run)
```

---

trimspace *Trim Whitespace*

Description

Trim leading and trailing whitespace from a character vector.

Usage

```
trimspace(charvec)
```

Arguments

charvec	A character vector to be trimmed.

Value

A character vector, the same length as `charvec`, without leading and trailing whitespace.

Examples

```
trimspace("    Bob   and Alice")
trimspace(" Harriet and    June    ")
```