Package 'shipunov'

April 1, 2019

Type Package

Title Miscellaneous Functions from Alexey Shipunov

Version 1.0

Date 2019-03-24

Author Alexey Shipunov [aut, cre], Karl W Broman [ctb], Paul Murrell [ctb], Marcello D'Orazio [ctb], Stephen Turner [ctb], Eugeny Altshuler [ctb], Roland Rau [ctb], Christoph Heibl [ctb], Marcus W Beck [ctb]

Maintainer Alexey Shipunov <dactylorhiza@gmail.com>

Description A collection of functions for data manipulation, plotting and statistical computing, to use separately or with the book ``Visual Statistics. Use R!'': Shipunov (2019) <http://ashipunov.info/shipunov/software/r/r-en.htm>.

Suggests PBSmapping, ape, class, cluster, effsize, gpclib, grid, ips, randomForest, nnet, scales, smirnov, vegan, MASS, adabag, e1071, neuralnet, rpart, tree

Imports methods

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2019-04-01 08:20:03 UTC

R topics documented:

`Adj.Rand` *Adjusted Rand index*

Description

Adjusted Rand index

Usage

```
Adj.Rand(cl1, cl2)
```

Arguments

`cl1`	Fisrt classification (character vector of group names)
`cl2`	Second classification

Details

Adjusted Rand Index.

Value

Similarity: numerical vector of length 1

Author(s)

Alexey Shipunov

References

Hubert L. and Arabie P. 1985. Comparing partitions. Journal of Classification. 2. 193–218.

See Also

Misclass

Examples

```
iris.dist <- dist(iris[, 1:4], method="manhattan")
iris.hclust <- hclust(iris.dist)
iris.3 <- cutree(iris.hclust, 3)
Adj.Rand(iris.3, iris[, 5])
```

Aggregate1 *Aggregates by one vector and uses it for row names*

Description

Aggregates by one vector and uses it for row names

Usage

```
Aggregate1(df, by, ...)
```

Arguments

df	Data frame to aggregate
by	Atomic object to use for aggregating
...	Arguments for 'aggregate()'

Details

'Aggregate1()' is an 'aggregate()' helper: aggregates only by one atomic variable and uses it for row names.

Value

Same as of 'aggregate()'

Author(s)

Alexey Shipunov

See Also

aggregate

Examples

```
trees3 <- sample(letters[1:3], nrow(trees), replace=TRUE)
Aggregate1(trees, trees3, median, na.rm=TRUE)
```

Alldups *Finds all duplicates*

Description

Finds duplicates from both ends, optionally returns indexes of duplicate groups

Usage

```
Alldups(v, groups = FALSE)
```

Arguments

v	vector
groups	If TRUE, uses as.numeric(as.character(v)) twice to index duplicated groups with natural numbers (and non-duplicated with 0)

Details

In short, this is extension of unique() which skips the first duplicate in each group. 'NA' count as duplicate but do not count as duplicate group.

Value

Logical vector of length equal to 'v', or numerical vector if 'groups=TRUE'

Author(s)

Alexey Shipunov

See Also

unique

Examples

```
aa <- c("one", "two", "", NA, "two", "three", "three", "three", NA, "", "four")
Alldups(aa)
data.frame(v=aa, dups=Alldups(aa), groups=Alldups(aa, groups=TRUE))
```

BootA *Bootstrap clustering*

Description

How to bootstrap clustering with 'ape'

Usage

```
BootA(dat, FUN=function(.x) ape::nj(dist(.x)), iter=1000, mc.cores=1, tresh=50,
 cons=TRUE, prop=0.5)
```

Arguments

dat	data
FUN	how to bootstrap (see examples)
iter	number of iterations, default 1000
mc.cores	how many cores to employ (system-dependent)
tresh	Treshold for printing bootstrap values
cons	Calculate consensus tree?
prop	0.5 is majority-rule consenus (default), 1 is strict consensus

Details

This is how to bootstrap clustering with 'ape::boot.phylo()'.

Author(s)

Alexey Shipunov

See Also

ape::boot.phylo

Examples

```
dat <- iris[, -5]
row.names(dat) <- abbreviate(make.names(iris[, 5], unique=TRUE))
iris.BA1 <- BootA(dat, iter=100)
plot(iris.BA1$boot.tree, show.node.label=TRUE)
plot(iris.BA1$cons.tree)
iris.BA2 <- BootA(dat, FUN=function(.x) ape::as.phylo(hclust(dist(.x))), iter=100)
## Not run:
## change (or remove) 'mc.cores=...' in accordance with your system features
iris.BA3 <- BootA(dat, FUN=function(.x) phangorn::NJ(dist(.x)), iter=100,
 mc.cores=4)

## End(Not run)
```

BootKNN *Bootstrap with kNN*

Description

How to bootstrap with kNN

Usage

```
BootKNN(data, classes, sub="none", nsam=4, nboot=1000, misclass=TRUE)
```

Arguments

data	Data frame to classify
classes	Character vector of class names
sub	Subsample to use (see example)
nsam	Number of training items from each level of grouping factor, default 4
nboot	Number of iterations
misclass	Calculate misclassification table?

Details

This is an example of how to bootstrap with 'class::knn1()'.

Samples equal numbers ('nsam') of training items from *each level* of grouping factor.

Allows to use *sub*set of data which will be used for subsampling of training data.

Value

Returns all predictions as character matrix, each boot is a column

Author(s)

Alexey Shipunov

See Also

class::knn1, Dev

Examples

```
iris.sub <- 1:nrow(iris) %in% seq(1, nrow(iris), 5)
iris.bootknn <- BootKNN(iris[, -5], iris[, 5], sub=iris.sub)
```

BootRF *Bootstrap with 'randomForest()'*

Description

How to bootstrap with 'randomForest()'

Usage

```
BootRF(data, classes, sub="none", nsam=4, nboot=1000, misclass=TRUE)
```

Arguments

data	Data frame to classify
classes	Character vector of class names
sub	Subsample to use (see example)
nsam	Number of training items from each level of grouping factor, default 4
nboot	Number of iterations
misclass	Calculate misclassification table?

Details

This an example of how to bootstrap with 'randomForest::randomForest()'.

Samples equal numbers ('nsam') of training items from *each level* of grouping factor.

Allows to use *sub*set of data which will be used for subsampling of training data.

Value

Returns all predictions as character matrix, each boot is a column

Author(s)

Alexey Shipunov

See Also

randomForest::randomForest, Dev

Examples

```
iris.sub <- 1:nrow(iris) %in% seq(1, nrow(iris), 5)

## could be slow
iris.bootrf <- BootRF(iris[, -5], iris[, 5], sub=iris.sub)
iris.bootrf <- BootRF(iris[, -5], iris[, 5])
## naturally, in the second case misclassification rate is lower
```

`Boxplots` *Grouped boxplots*

Description

Boxplots for every scaled variable grouped by factor

Usage

```
Boxplots(vars, groups, boxcols=Pastels, legpos="topleft", srt=45, adj=1,
 slty=3, yticks=FALSE, ymarks=FALSE, ...)
```

Arguments

`vars`	data frame consists of variables to plot
`groups`	grouping factor
`boxcols`	colors of character boxes, default is 'Pastels', i.e. c("white", "lightblue", "mistyrose", "lightcyan", "lavender", "cornsilk")
`legpos`	where to place automatic legend, default is 'topleft', for no legend use 'legpos=NA'
`slty`	line type to delimit groups of boxes
`srt, adj, yticks, ymarks`	regular 'plot()' arguments
`...`	additional arguments to 'boxplot()'

Details

There are many ways to represent groups in data. One is trellis plots. 'Boxplots()' make grouped plots which fit the plot box linearly and therefore easy to compare. So the main idea for grouped plots is to make comparison easier.

Please note that because characters within group are likely of different nature, they are scaled. Consequently, tick marks are removed as they have no sense.

Value

For the efficiency reasons, the function does not return anything.

Author(s)

Alexey Shipunov

See Also

boxplot, Linechart, Dotchart3

Examples

```
Trees <- trees
Trees[, 4] <- sample(letters[1:3], nrow(Trees), replace=TRUE)
Boxplots(Trees[, 1:3], factor(Trees[, 4]), srt=0, adj=c(.5, 1)) # horizontal labels

lo <- read.table("http://ashipunov.info/shipunov/open/eq_l.txt", h=TRUE, sep=";")
me <- read.table("http://ashipunov.info/shipunov/open/eq_s.txt", h=TRUE, sep=";")
## uses Recode() from this same package
sp <- Recode(me$N.POP, lo$N.POP, as.character(lo$SPECIES))
eq <- cbind(sp, me[-1])
eq3 <- eq[eq$sp %in% levels(eq$sp)[1:3], ]
Boxplots(eq3[, 2:9], eq3[, 1], boxcols=grey(1:3/3), slty=0) # no border lines
```

Cdate *System date, time plus easy save history*

Description

System date in 'yyyymmdd' format, system time in 'yyyymmdd_hhmmss' format plus easy save history

Usage

```
Cdate()
Ctime()
Save.history()
```

Details

System date / time in compact formats. These formats are by experience, the most appropriate formats both for file systems and for spreadsheets.

There is also easy 'savehistory' (does not work under macOS R GUI – but works under macOS Terminal.app R).

Author(s)

Alexey Shipunov

See Also

savehistory

Examples

```
Cdate()
Ctime()
## Not run:
## does not work under macOS GUI
Save.history()

## End(Not run)
```

Cladd *Adds confidence bands to the simple linear model plots*

Description

Adds confidence bands to the simple linear model plots

Usage

```
Cladd(model, data, level=.95, lty=2, ab.lty=0, col="black", ab.col="black")
```

Arguments

model	Simple linear model name
data	Original data
level	Confidence level
lty	Confidence bands line type
ab.lty	Regression line type
col	Confidence bands line color
ab.col	Regression line color

Details

'Cladd()' adds confidence bands to the simple linear model plots. Works only for simple lm(y ~ x) objects!

Author(s)

Alexey Shipunov

See Also

lm

Examples

```
hg.lm <- lm(Height ~ Girth, data=trees)
plot(Height ~ Girth, data=trees)
Cladd(hg.lm, data=trees, ab.lty=1)
```

Co.test *Correlation test between cophenetic and original distances*

Description

Correlation test between cophenetic and original distances

Usage

```
Co.test(hclust, dist, method="spearman")
```

Arguments

hclust	Hclust object
dist	Distance matrix
method	How to calculate correlation

Details

Correlation between cophenetic distances and original distances.

Reveals the consistency of used methods.

Spearman correlation is default because cophenetic distances are frequently non-parametric.

Author(s)

Alexey Shipunov

Examples

```
iris.d <- dist(iris[, -5])
iris.h <- hclust(iris.d)
Co.test(iris.h, iris.d, method="kendall")
```

Coeff.det *Average coefficients of determination for each variable*

Description

Average coefficients of determination for each variable

Usage

```
Coeff.det(X, ...)
```

Arguments

X	Data frame or matrix with values
...	Arguments to 'cor()'

Details

Average coefficients of determination for each variable.

Allow to compare various correlation structures (Rostova, 1999).

Value

Numerical vector of coefficients of determination

Author(s)

Alexey Shipunov

References

Rostova N.S. 1999. The variability of correlations systems between the morphological characters. Part 1. Natural populations of Leucanthemum vulgare (Asteraceae). Botanicheskij Zhurnal. 84(11): 50–66.

Examples

```
Coeff.det(trees, use="pairwise")
```

Coml *Compare checklists*

Description

Compare species checklists

Usage

```
Coml(df1, df2)
## S3 method for class 'Coml'
summary(object, ..., n=10)
```

Arguments

df1	First data frame with species presence/absence data, species as row names
df2	Second data frame
object	Object of the class 'Coml'
n	Number of indicator species
...	Additional arguments

Details

Compare two (groups of) checklists (Abramova et al., 2003).

Calculates difference (in %) between checklists with *common base*, i.e., species occurrence/abundance columns of data frame with species names as row names.

Finds names of "indicators" most characteristic to each group

Value

Object of the class 'Coml', or nothing

Author(s)

Alexey Shipunov

References

Abramova L. A., Rimskaya-Korsakova N. N., Shipunov A. B. 2003. The comparative study of the flora of Kiv Gulf, Chupa Gulf and Keret' Archipelago islands (Kandalaksha Bay of White Sea). Proceedings of the Pertsov White Sea Biological Station. Vol. 9. Moscow. P. 22–33. in Russian (English abstract)

Examples

```
y <- read.table("http://ashipunov.info/shipunov/open/dolbli.txt", h=TRUE, sep="\t")
y.Coml <- Coml(y[1:45], y[46:79])
summary(y.Coml, n=5)
```

Cor *Correlation matrix with p-values*

Description

Correlation matrix with p-values

Usage

```
Cor(X, stars=TRUE, dec=4, p.level=0.05, ...)
Cor2(X, dec=4, p.level=0.05)
```

Arguments

X	Matrix or data frame with values
stars	Replaces p-values with stars if it not greater than 'p.level'
dec	Decimal point
p.level	P-level
...	Arguments to 'cor.test()'

Details

'Cor()' calculates correlation matrix with p-values.

'Cor2()' is another (faster) variant of correlation matrix with p-values based on F-statistic. Shows significances in the upper triagle. Uses Pearson correlation only but much faster than 'Cor()'.

Author(s)

Alexey Shipunov

Examples

```
Cor(longley, dec=2)
Cor2(longley, dec=2)
```

Cor.vec *Calculates correlation and converts results into the named long vector*

Description

Calculates correlation and converts results into the named long vector

Usage

```
Cor.vec(X, ...)
```

Arguments

X	Data frame or matrix with values
...	Arguments to 'cor()'

Details

Calculates correlation and converts results into the named long vector.

Value

Named numerical vector of correlations.

Author(s)

Alexey Shipunov

References

Rostova N.S. 1999. The variability of correlations systems between the morphological characters. Part 1. Natural populations of Leucanthemum vulgare (Asteraceae). Botanicheskij Zhurnal. 84(11): 50–66.

See Also

Rostova.tbl

Examples

```
Cor.vec(trees, method="spearman")
```

CVs *Coefficients of variation*

Description

Coefficients of variation

Usage

```
CVs(sample, na.rm=TRUE)
```

Arguments

sample	Numerical vector
na.rm	Remove NAs?

Details

Coefficients of variation: different variants of the standardized range

Value

Named numerical vector

Author(s)

Alexey Shipunov

Examples

```
sapply(trees, CVs)
```

Dev *Which object is predicted with less accuracy?*

Description

Allows to know which object is predicted with less accuracy

Usage

```
Dev(pred, useNA = "no", adj = FALSE)
```

Arguments

pred	Pedictions as character matrix, each boot is a column
useNA	Use NAs?
adj	Adjust deviation coefficients? (see below)

Details

Allows to know which object is predicted with less accuracy.

It calculates deviation coeffitient out of the table where rows are objects, and columns are (bootstrapped) predictions.

By default, does not use NA's (use useNA="always" on your risk).

Deviation coefficient is a minimal absolute deviation from one of the range ends (0 and number of predictions), divided by number of predictions, multiplied by number of deviations minus one, and (optionally) adjusted by division to number of predition levels (to make different situations comparable).

Value

Numeric vector of deviation coefficients

Author(s)

Alexey Shipunov

See Also

BootRF, BootKNN

Examples

```
## could be slow
iris.bootrf <- BootRF(iris[, -5], iris[, 5])
data.frame(Iris=make.names(iris[, 5], unique=TRUE), Dev=Dev(iris.bootrf))
```

Dotchart *Improved dotcharts*

Description

Dotcharts, improved and extended

Usage

```
Dotchart1(x, labels=NULL, groups=NULL, gdata=NULL, cex=par("cex"), pt.cex=cex,
 pch=21, gpch=21, bg=par("bg"), color=par("fg"), gcolor=par("fg"), lcolor="gray",
 xlim=range(x[is.finite(x)]), main=NULL, xlab=NULL, ylab=NULL, ...)

Dotchart(...)

Dotchart3(values, left, right, pch=21, bg="white", pt.cex=1.2, lty=1, lwd=2,
 gridcol="grey", ...)
```

Arguments

x	Either a vector or matrix of numeric values. Inputs are coerced by 'as.numeric()', with a message.
labels	A vector of labels for each point.
groups	An optional factor indicating how the elements of 'x' are grouped.
gdata	Data values for the groups. This is typically a summary such as the median or mean of each group.
cex	The character size to be used.
pt.cex	The 'cex' to be applied to plotting symbols.
pch	The plotting character or symbol to be used.
gpch	The plotting character or symbol to be used for group values.
bg	The background color of plotting characters.
color	The color(s) to be used for points and labels.
gcolor	The single color to be used for group labels and values.
lcolor	The color(s) to be used for the horizontal lines.
xlim	Horizontal range for the plot.
main	Overall title for the plot, see 'title'.
xlab, ylab	Axis annotations as in 'title'.
values	Centers for 'Dotchart3()'
left	Left margins for 'Dotchart3()'
right	Right margins for 'Dotchart3()'
lty	Line type for 'Dotchart3()'
lwd	Line width for 'Dotchart3()'
gridcol	Grid color for 'Dotchart3()'
...	Additional arguments

Details

For better explanations of options, see 'help(dotchart)'.

'Dotchart1()' is a default 'dotchart()' corrected for use with 1-dimensional tables with 'ylab' and/or changed 'par("mar")[2]'. So comparing with the default 'dotchart()', it has a better left margin. It also outputs message instead of warning.

'Dotchart()' is a prettified 'Dotchart1()' with the following defaults: 'Dotchart1(lcolor="black", bg="white", pt.cex=1.2, ...)'. Use it as a shortcut.

'Dotchart3()' shows values together with ranges. It is an extension of 'Dotchart1()'; for each value, it shows ranges. Does not work with grouped data. A bit similar to 'Linechart()' but more general.

Author(s)

Alexey Shipunov

See Also

dotchart, Linechart

Examples

```
## Compare:
aa <- table(c(1, 1, 1, 2, 2, 3))
##
Dotchart1(aa, ylab="Ylab") # shows ylab; outputs message instead of warning
dotchart(aa, ylab="Ylab") # does not show ylab; produces warning
##
## changes all margins (note: Dotchart1() cannot forcedly _decrease_ left margin)
old.par <- par(mar=c(1, 10, 1, 1)) ; Dotchart1(aa, ylab="Ylab") ; par(old.par)
## does not change left margin
old.par <- par(mar=c(1, 10, 1, 1)) ; dotchart(aa, ylab="Ylab") ; par(old.par)

Dotchart(aa)

Dotchart3(structure(1:3, names=LETTERS[1:3]), 0:2, 2:4)
Dotchart3(structure(0:2, names=LETTERS[1:3]), 0:2, 2:4, pch="") # ranges only
```

Draw.arrow *Add arrow to the plot interactively*

Description

Add arrow to the plot interactively

Usage

```
Draw.arrow(reverse=FALSE, horizontal=FALSE, vertical=FALSE, length=0.1, ...)
```

Arguments

reverse	If FALSE, first indicate the tail of the arrow and then the head; if TRUE, first indicate the head of the arrow and then the tail.
horizontal	If TRUE, force the arrow to be horizontal; use the average y-axis value of the two clicks for the vertical placement.
vertical	If TRUE, force the arrow to be vertical; use the average x-axis value of the two clicks for the horizontal placement.
length	Length of the edges of the arrow head.
...	arguments to 'arrows()'

Details

'Draw.arrow()' adds arrow to the plot interactively. Based on analogous Karl W Broman function. Uses the 'locator()' function to plot an arrow.

Value

The locations of the endpoints of the arrow, as a two-row matrix. The first row indicates the location of the tail of the arrow; the second row indicates the location of the head of the arrow.

Author(s)

Alexey Shipunov

See Also

locator

Examples

```
plot(1:20)
## Not run:
## interactive command:
Draw.arrow(col="red", lwd=2)

## End(Not run)
```

Ell *Plot ellipse*

Description

Plot ellipse

Usage

```
Ell(x, y, width, height=width, theta=2*pi, npoints=100, plot=TRUE, ...)
```

Arguments

`x`	x coordinate of center
`y`	y coordinate of center
`width`	length of major axis
`height`	length of minor axis
`theta`	rotation
`npoints`	number of points to send to polygon
`plot`	if TRUE, add to current device, if FALSE, returns list of components
`...`	arguments to 'polygon()'

Details

Plots ellipse based on 'polygon()'.

Value

If plot=FALSE, returns list of components.

Author(s)

Alexey Shipunov

Examples

```
plot(1:8, type="n")
Ell(4, 5, 6)
```

`Ellipses` *Groups' confidence ellipses*

Description

Groups' confidence ellipses

Usage

```
Ellipses(pts, groups, match.color=TRUE, usecolors=NULL, centers=FALSE,
 c.pch=0, c.cex=3, level=0.95, ...)
```

Arguments

pts	Data points to plot
groups	Grouping variable (mumerical)
level	Confidence level
match.color	Match colors
usecolors	Use colors (palette)
centers	Show centrs?
c.pch	Color of center points
c.cex	Scale of center points
...	Arguments to 'Confelli()' and finally to 'plot()'

Details

Internally, uses 'Confelli()' function which plots an ellipse with covariance matrix C, center b, and P-content level according the F(2, df) distribution.

Author(s)

Alexey Shipunov

See Also

Hulls

Examples

```
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
plot(iris.p, type="n", xlab="PC1", ylab="PC2")
text(iris.p, labels=abbreviate(iris[, 5], 1, method="both.sides"))
Ellipses(iris.p[, 1:2], as.numeric(iris[, 5]), centers=TRUE)
```

Ex.boxplot *Boxplot explanation*

Description

Boxplot explanation

Usage

```
Ex.boxplot(...)
```

Arguments

...	Arguments to 'boxplot()'

Details

The scheme which explains typical boxplot.

Author(s)

Alexey Shipunov

See Also

boxplot

Examples

```
Ex.boxplot()
```

Ex.col *Examples of colors*

Description

Examples of standard colors (also shows all colors)

Usage

```
Ex.col(all=FALSE)
Ex.cols(all=FALSE)
```

Arguments

all	if TRUE, shows all 476 named colors

Details

Examples of standard colors (also shows all colors). For the palettes, run 'example(rainbow)'.

Note: large device is required to see all 476 named colors.

Author(s)

Alexey Shipunov

See Also

palette

Examples

```
Ex.cols()
Ex.cols(all=TRUE)
```

Ex.font *Examples of fonts*

Description

Examples of standard fonts

Usage

```
Ex.font()
```

Details

Examples of standard fonts

Author(s)

Alexey Shipunov

See Also

par

Examples

```
Ex.fonts()
```

Ex.lty *Examples of line types*

Description

Line type examples

Usage

```
Ex.lty(custom="431313")
Ex.lines(custom="431313")
```

Arguments

custom	character string to specify custom line type (see '?lines').

Details

Line type examples. To see other possible custom line types, try custom="F8" or similar.

Author(s)

Alexey Shipunov

See Also

lines

Examples

```
Ex.lines(custom="F8")
```

`Ex.margins` *Example of plot margins*

Description

Example of plot margins

Usage

```
Ex.margins()
```

Details

Example of plot margins. Modified from Paul Murrell (2006).

Author(s)

Alexey Shipunov

References

Murrell P. 2006. R Graphics.

See Also

par

Examples

```
Ex.margins()
```

Ex.pch *Point examples*

Description

Point ('pch') examples

Usage

```
Ex.pch(extras=c("*", ".", "+", "a"), cex=2, col="black", bg="gray",
 coltext="black", cextext=1.2, main="")
Ex.points(extras=c("*", ".", "+", "a"), cex=2, col="black", bg="gray",
 coltext="black", cextext=1.2, main="")
```

Arguments

extras	which extra symbols to show
cex	point scale, default 2
col	point color, default black
bg	point background (for symbols with a 'bg'-colored interior), default grey
coltext	text color, default black
cextext	text scale, default 1.2
main	plot title, no title by default

Details

Point ('pch') examples, modified from 'example(points)'.

Author(s)

Alexey Shipunov

See Also

points

Examples

```
Ex.points()
```

Ex.plots *Examples of plot types*

Description

Examples of plot types

Usage

```
Ex.plots()
Ex.types()
```

Details

Examples of nine standard plot types.

Author(s)

Alexey Shipunov

See Also

par

Examples

```
Ex.types()
```

Files *Textual filesystem browser*

Description

Textual filesystem browser

Usage

```
Files(root=getwd(), multiple=FALSE, hidden=FALSE)
```

Arguments

root	Root directory
multiple	Allows multiple files to be selected
hidden	Show hidden files?

Details

Interactive text-based file chooser dialog. Alternatives for Linux: tcltk::tk_choose.files() and tcltk::tk_choose.dir()

Value

Returns character vector of selected files, or dirname (useful for 'setwd()'), or new user-defined file name with full path.

Author(s)

Alexey Shipunov

See Also

setwd, getwd, dir

Examples

```
## interactive commands
setwd <- Files() # then select directory to work in
Files("~", hidden=TRUE) # explore home directory with hidden files (Linux, macOS)
```

Gap.code *Gap coding*

Description

Gap coding of DNA nucleotide alignments

Usage

```
Gap.code(seqs)
```

Arguments

seqs	Character vector of aligned (and preferably flank trimmed) DNA sequences.

Details

FastGap-like gap code nucleotide alignments ('ATGCN-' are allowed).

Encodes gap presence as 'A' and absence as 'C'.

Likely too straightforward, and only weakly optimized (really slow).

Author(s)

Alexey Shipunov

References

Borchsenius F. 2009. FastGap 1.2. Department of Biosciences, Aarhus University, Denmark. Published online at <http://www.aubot.dk/FastGap_home.htm>

Examples

```
write(file=file.path(tempdir(), "tmp.fasta"),  c(
 ">1\nGAAC------ATGC",
 ">2\nGAAC------TTGC",
 ">3\nGAAC---CCTTTGC",
 ">4\nGAA---------GC"))
write(file=file.path(tempdir(), "tmp_expected.fasta"), c(
 ">1\nGAAC------ATGCCA-",
 ">2\nGAAC------TTGCCA-",
 ">3\nGAAC---CCTTTGCCCA",
 ">4\nGAA---------GCA--"))
tmp <- Read.fasta(file=file.path(tempdir(), "tmp.fasta"))
expected <- Read.fasta(file=file.path(tempdir(), "tmp_expected.fasta"))
seqs <- tmp$sequence
gc <- Gap.code(seqs)
tmp$sequence <- apply(cbind(seqs, gc), 1, paste, collapse="")
identical(tmp, expected) # TRUE, isn't it?
```

Gower.dist *Gower distance*

Description

Calculates Gower distance

Usage

```
Gower.dist(data.x, data.y=data.x, rngs=NULL, KR.corr=TRUE)
```

Arguments

data.x	A matrix or a data frame containing variables that should be used in the computation of the distance.
data.y	A numeric matrix or data frame with the same variables, of the same type, as those in data.x
rngs	A vector with the ranges to scale the variables. Its length must be equal to number of variables in data.x
KR.corr	When TRUE (default) the extension of the Gower's dissimilarity measure proposed by Kaufman and Rousseeuw (1990) is used. Otherwise the original Gower's (1971) formula is considered.

Details

Gower.dist() code was taken (and slightly amended to keep dimnames and return 'dist' object in case of one matrix) from 'StatMatch' package; please see this package for the original code and full documentation.

This function computes the Gower's distance (dissimilarity) among units in a dataset or among observations in two distinct datasets. Columns of mode numeric will be considered as interval scaled variables; columns of mode character or class factor will be considered as categorical nominal variables; columns of class ordered will be considered as categorical ordinal variables and, columns of mode logical will be considered as binary asymmetric variables. Missing values (NA) are allowed. If only data.x is supplied, the dissimilarities between _rows_ of data.x will be computed.

For 'rngs', in correspondence of non-numeric variables, just put 1 or NA. When rngs=NULL (default) the range of a numeric variable is estimated by jointly considering the values for the variable in 'data.x' and those in 'data.y'.

Value

A matrix object with distances among rows of data.x and those of data.y

Author(s)

Marcello D'Orazio, <mdo.statmatch@gmail.com>

References

Gower J.C. 1971. A general coefficient of similarity and some of its properties. Biometrics, 27, 623–637.

Kaufman L., Rousseeuw P.J. 1990. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York.

See Also

dist, cluster::daisy

Examples

```
x1 <- as.logical(rbinom(10, 1, 0.5))
x2 <- sample(letters, 10, replace=TRUE)
x3 <- rnorm(10)
x4 <- ordered(cut(x3, -4:4, include.lowest=TRUE))
xx <- data.frame(x1, x2, x3, x4, stringsAsFactors=FALSE)

## matrix of distances among first obs. in xx and the remaining ones
Gower.dist(data.x=xx[1:6, ], data.y=xx[7:10, ])

## matrix of distances among observations in xx
row.names(xx) <- LETTERS[1:nrow(xx)]
dx <- Gower.dist(xx)
plot(hclust(dx))
```

Gradd *Classification grid*

Description

Adds to the 2D ordination plot small semi-transparent points which make color classification grid

Usage

```
Gradd(model2var, data2var, spacing=75, trnsp=0.3, pch=20, cex=0.2,
 palette=NULL, type="ids", ...)
```

Arguments

model2var	model based on data2var
data2var	data with exactly 2 variables
spacing	space between points
trnsp	transparency
pch	type of point
cex	scale of point
palette	palette to use
type	type of the model: "ids", "lda", "neuralnet", "tree", or "user" (see examples)
...	arguments to 'plot()'

Details

'Gradd()' adds to the 2D ordination plot small semi-transparent points which make color classification grid.

Requires model with 'predict' method to be computed first.

Model should use ids (to make colors) and exactly 2 variables with names same as data2var column names, e.g:

model2var <- somefunction(ids ~ ., data=cbind(ids, data2var))

If type="user", uses predefined 'User.Predict(model2var, X)' function which must return factor ids from testing X data.

Please see examples to understand all of these better.

Note that instead of dots, one can use contours, but they are harder to employ because they need membership values in order to calculate borders (places where memberships are equal).

Author(s)

Alexey Shipunov

Examples

```
## SVM:
library(e1071)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.svm.pca <- svm(Species ~ ., data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="SVM")
Gradd(iris.svm.pca, iris.p)
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## LDA:
library(MASS)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.lda.pca <- lda(Species ~ . , data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="LDA")
Gradd(iris.lda.pca, iris.p, type="lda")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## 'tree::tree':
library(tree)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.tree.pca <- tree(Species ~ . , data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="tree")
Gradd(iris.tree.pca, iris.p, type="tree")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## randomForest:
library(randomForest)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.rf.pca <- randomForest(Species ~ ., data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="randomForest")
Gradd(iris.rf.pca, iris.p)
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## naiveBayes:
library(e1071)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.nb.pca <- naiveBayes(Species ~ ., data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="naiveBayes")
Gradd(iris.nb.pca, iris.p)
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## neuralnet:
library(neuralnet)
iris.p2 <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.p2 <- cbind(iris.p2, Tobin(iris$Species, convert.names=FALSE))
iris.nn.pca <- neuralnet(setosa + versicolor + virginica ~ PC1 + PC2, data=iris.p2,
 hidden=3, lifesign="full")
```

```
plot(iris.p, type="n", main="neuralnet")
Gradd(iris.nn.pca, iris.p, type="neuralnet")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## rpart + MDS for the base plot:
iris.dist <- dist(iris[, 1:4], method="manhattan")
iris.dist[iris.dist == 0] <- abs(jitter(0))
library(MASS)
iris.m <- isoMDS(iris.dist)$points
colnames(iris.m) <- c("Dim1", "Dim2")
library(rpart)
iris.rpart.mds <- rpart(Species ~ . , data=cbind(iris[5], iris.m))
plot(iris.m, type="n", main="rpart + MDS")
Gradd(iris.rpart.mds, iris.m, type="tree")
text(iris.m, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## QDA:
library(MASS)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.qda.pca <- qda(Species ~ . , data=cbind(iris[5], iris.p))
plot(iris.p, type="n", main="QDA")
Gradd(iris.qda.pca, iris.p, type="lda")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## AdaBoost:

library(adabag)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.ada.pca <- boosting(Species ~ . , data=cbind(iris[5], iris.p)) # slow!
plot(iris.p, type="n", main="AdaBoost")
Gradd(iris.ada.pca, iris.p, type="lda")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))

##
## kNN:
library(class)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
plot(iris.p, type="n", main="kNN")
User.Predict <- function(model2var, X) knn(train=model2var[, 2:3], test=X,
 cl=model2var[, 1], k=5)
Gradd(cbind(iris[5], iris.p), iris.p, type="user")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
## nnet:
library(nnet)
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
iris.nnet.pca <- nnet(Species ~ . , data=cbind(iris[5], iris.p), size=4)
plot(iris.p, type="n", main="nnet")
```

```
Gradd(iris.nnet.pca, iris.p, type="tree")
text(iris.p, col=as.numeric(iris[, 5]), labels=abbreviate(iris[, 5], 1,
 method="both.sides"))
##
```

Gridmoon *Draw with 'R'*

Description

Sraw with 'R'

Usage

```
Gridmoon(Skyres=50, Nightsky=TRUE, Daysky="deepskyblue", Moon=TRUE,
 Moonsize=0.05, Stars=TRUE, Hillcol="black", Text=c("Once upon a time..."),
 Textsize=22, Textpos=c(.15, .51), Textcol="white")
```

Arguments

Skyres	Sky resoluiton
Nightsky	If TRUE, ther eis a night
Daysky	Color of day sky
Moon	If TRUE, there is a moon
Moonsize	Moon size
Stars	If TRUE, there are stars
Hillcol	Hill color
Text	Text to print
Textsize	Text size
Textpos	Text position
Textcol	Text color

Details

'Gridmoon()' is an example how to paint (draw) with 'R'. Just for fun. From Murrell (2006) "R Graphics", with modifications.

Author's comments:

An example of a one-off image drawn using the grid system.

The code is somewhat modular and general, with functions for producing different shapes, but the sizes and locations used in this particular image assume a 2:1 aspect ratio.

The gradient-fill background (dark at the top to lighter at the bottom) is achieved by filling multiple overlapping polygons with slowly changing shades of grey.

Author(s)

Alexey Shipunov

References

Murrell P. 2006. R Graphics.

Examples

```
## Examples best viewed with 2:1 aspect ratio
Gridmoon(Skyres=75)
Gridmoon(Nightsky=FALSE, Moon=FALSE, Stars=FALSE, Hillcol="forestgreen",
 Text="Use R!", Textcol="yellow", Textpos=c(.25, .85), Textsize=96)
```

Hclust.match *Counts matches between two hierarchical clusterings*

Description

Counts matches between two hierarchical clusterings

Usage

```
Hclust.match(hc1, hc2, scale=FALSE)
```

Arguments

hc1	First hclust object
hc2	Second hclust object
scale	Scale by the sum size of trees?

Details

'Hclust.match()' counts matches between two hierarchical clusterings (based on 'cutree()').

Result is a sort of consensus distances. Useful, for example, for clustering heatmaps.

Author(s)

Alexey Shipunov

Examples

```
aa <- read.table("http://ashipunov.info/data/atmospheres.txt", h=TRUE, sep="\t", row.names=1)
aa.d1 <- hclust(dist(t(aa)))
aa.d2 <- hclust(as.dist(1 - abs(cor(aa, method="s"))), method="ward.D")
aa12.match <- Hclust.match(aa.d1, aa.d2)
heatmap(aa12.match, scale="none")
```

Histp *Histogram with percents*

Description

Histogram with percents on the top of bars

Usage

```
Histp(x, breaks, ...)
```

Arguments

x	numeric vector
breaks	number of breaks
...	other arguments to 'hist()'

Details

Histp plots histogram with percents on the top of bars, it is useful to know exactly the proportion of observations in each bin

Author(s)

Alexey Shipunov

See Also

hist

Examples

```
Histp(iris[, 1])
```

Histr *Histogram with overlaid curve*

Description

Histogram with overlaid normal curve or density, optionally with rug

Usage

```
Histr(x, overlay="normal", rug=FALSE, col="gray80", ...)
```

Arguments

x	numerical vector
overlay	type of curve to overlay, accepted values are "normal" and "density"
rug	if TRUE, will add rug plot
col	curve color
...	arguments to 'hist()'

Details

Histr plots histogram with overlaid normal curve or density, optionally with rug. Modified from Stephen Turner's 'Tmisc' package.

Author(s)

Alexey Shipunov

See Also

hist, density, rnorm

Examples

```
x <- rnorm(1000, mean=5, sd=2)
Histr(x)
Histr(x, overlay="density")
Histr(x^2, overlay="density", rug=TRUE, breaks=50, col="lightblue2")
```

Hulls *Groups' hulls*

Description

Groups' hulls with centroids calculation

Usage

```
Hulls(pts, groups, match.color=TRUE, usecolors=NULL, plot=TRUE, centers=FALSE,
 c.pch=0, c.cex=3, ...)
```

Arguments

pts	Data points to plot
groups	Grouping variable (mumerical)
match.color	Match color
usecolors	Use colors (palette)
plot	Plot?
centers	Show centers?
c.pch	Color of center points
c.cex	Scale of center points
...	Arguments to 'lines()'

Details

Groups' hulls with optoinal centroids calculation (requires 'PBSmapping')

Value

Invisibly outputs list of hulls with coordinates

Author(s)

Alexey Shipunov

See Also

Ellipses

Examples

```
iris.p <- prcomp(iris[, 1:4], scale=TRUE)$x[, 1:2]
plot(iris.p, type="n", xlab="PC1", ylab="PC2")
pal <- rainbow(3)
text(iris.p, labels=abbreviate(iris[, 5], 1, method="both.sides"),
 col=pal[as.numeric(iris[, 5])])
Hulls(iris.p[, 1:2], as.numeric(iris[, 5]), centers=TRUE, usecolors=pal)
```

`Infill` *Rarefaction curves*

Description

Rarefaction curves

Usage

```
Infill(x, n=10)
## S3 method for class 'Infill'
plot(x, ...)
## S3 method for class 'Infill'
summary(object, ...)
```

Arguments

`x`	Data frame where columns are species
`object`	Object of the class "Infill"
`n`	Number of permutations
`...`	Arguments to 'plot()' or 'summary()'

Details

'Infill()' returns matrix to draw accumulation curves (each column is one curve).

'Infill' uses checklists of biological organisms to build rarefaction curves. You can estimate how many taxa will appear in the next sample to plan your investigations (e.g. revealing flora or fauna of the certain area).

If cells contain taxa abundance it will be automatically replaced with 1 or 0. Permutation is a random shuffle of the samples to get more valid estimation of the taxa accumulation process. It does not matter which sample appeared first. The resulting plot gives information on the process of taxa revealing during the investigation. High number of permutations gives more presize results, but the calculations are more slow. Empirically, 100 permutations are enough. The plot indicates full taxa number which has been accumulated in this and all the previous samples.

Value

Object of the class "Infill", or nothing

Author(s)

Alexey Shipunov, Eugeny Altshuler

References

Diaz-Frances E., Soberon J. 2005. Statistical estimation and model selection of species accumulation curves. Conservation Biology. Vol. 19, N 2. P. 569-573.

Gotelli N.J., Colwell R.C. 2001. Quantifying biodiversity: procedures and pitfalls in the measurement and comparison of species richness. Ecology Letters. Vol. 4. P. 379-391.

Soberon J.M., Llorente J.B. 1993. The use of species accumulation functions for the prediction of species richness. Conservation Biology. Vol. 7. N 3. P. 480-488.

Examples

```
x <- t(read.table("http://ashipunov.info/shipunov/open/dolbli.txt", h=TRUE, sep="\t"))
data <- x[1:45, ] # one of two lakes selected
data.I <- Infill(data)
summary(data.I)
plot(data.I)
```

Is.tax.inform.char *Taxonomic informativeness*

Description

Is the character potentially taxonomically informative?

Usage

```
Is.tax.inform.char(vec)
```

Arguments

vec	Character vector from the column of DNA alignment

Details

Is the character potentially taxonomically informative?

If DNA encoding used, 'Nn?' should be converted into NA. Gaps ('-') counted, NAs not.

Value

Number of the potentially taxonomically informative characters.

Author(s)

Alexey Shipunov

Examples

```
Is.tax.inform.char(c("A", "C", "T", "T", "T", "T", "A"))
Is.tax.inform.char(c("A", "C", "T", "T", "T", "T", "G"))
Is.tax.inform.char(c("-", "T", "T", "T", "T", "T", "-", NA))
Is.tax.inform.char(c("A", "T", "T", "T", "T", "T", NA, NA))
Is.tax.inform.char(c(1, 0, 1, 1, 0, 0, 0, 0))
```

Jclust *Simple bootstrap and jackknife clustering*

Description

Simple bootstrap and jackknife clustering

Usage

```
Jclust(data, n.cl, iter=100, method.d="manhattan", method.c="ward.D", bootstrap=TRUE)
## S3 method for class 'Jclust'
print(x, ...)
## S3 method for class 'Jclust'
plot(x, main="", xlab="", sub=NULL, rect.lty=3, rect.col=1, ...)
```

Arguments

data	Data
n.cl	Number of desired clusters
iter	Number of iterations
method.d	Distance method
method.c	Hierarchical clustering method
bootstrap	Bootstrap or jacknife?
x	Object of the class 'Jclust'
rect.lty	Line type for the rectangles
rect.col	Color of rectangles
main	Plot title
xlab	Horizontal axis label
sub	Horizontal axis sub-label
...	Additional arguments to the 'print()' or 'plot.hclust()'

Details

Simple bootstrap and jackknife clustering, requires the desired number of clusters.

This is how you can boostrap *all* reliable cluster numbers:

'for (i in 2:(nrow(t(mo)) - 1)) print(J.Clust(t(mo), i, iter=1000, boot=TRUE))'

Alternatives: 'pvc::pvc()'; 'fpc::clusterboot()'; 'ClassDiscovery::BootstrapClusterTest()'

Author(s)

Alexey Shipunov

Examples

```
mo <- read.table("http://ashipunov.info/shipunov/open/moldino.txt", h=TRUE, sep="\t", row.names=1)
(mo.j <- Jclust(t(mo), 3, iter=1000))
plot(mo.j)
```

K *Coefficient of divergence*

Description

Lyubishchev's coefficient of divergence (SSMD^2)

Usage

```
K(x, y=NULL, data=NULL, mad=FALSE, na.rm=TRUE)
## S3 method for class 'K'
print(x, ...)
## S3 method for class 'K'
summary(object, ..., num=2)
```

Arguments

x	Numeric vector, or formula, or object of the class 'K'
y	Second numeric vector, or nothing
data	Data with two columns (in case of formula)
mad	Non-parametric variant of K (not Lyubishchev's)
na.rm	Remove NAs?
object	Object of the class 'K'
num	Digits to round
...	Additional arguments

Details

One of the effect size measures, Lyubishchev's K, coefficient of divergence (Lyubishchev, 1959).
Interestingly, the recently invented "striclty standardized mean difference" SSMD ('https://en.wikipedia.org/wiki/Strictly_sta
is just a square root of K.

Value

Value of K, or nothing.

Author(s)

Alexey Shipunov

References

Lyubishchev A. A. 1959. How to apply biometry to systematics. Leningrad University Herald. N 9. P. 128–136. [In Russian, English abstract].

Examples

```
K(1:3, 2:100)
eq2 <- read.table("http://ashipunov.info/data/eq.txt", h=TRUE)
sapply(eq2[, -1], function(.x) K(.x ~ eq2[, 1]))
```

Life *Game of Life*

Description

Conway's Game of Life

Usage

```
Life(n.rows=100, n.cols=100, n.cycles=100, sleep.time=.1, cols=c("#f0f0f0", "#2f81c1"),
 rnd.threshold=0.3)
```

Arguments

`n.rows`	number of rows
`n.cols`	number of columns
`n.cycles`	number of cycles
`sleep.time`	time for pause after each cycle
`cols`	main colors
`rnd.threshold`	0 empty board; 1 all squares are filled

Details

Please note that at the moment, the function is not interactive.

Author(s)

Alexey Shipunov

Examples

```
Life(10, 10, 10, .3)
```

`Linechart` *Dotchart-like plot sfor every scaled variable grouped by factor*

Description

Dotchart-like plot sfor every scaled variable grouped by factor

Usage

```
Linechart(vars, groups, xticks=TRUE, xmarks=TRUE, mad=FALSE, pch=19,
 se.lwd=1, se.col=1, ...)
```

Arguments

`vars`	Variables to draw (data frame)
`groups`	Grouing factor
`xticks`	Show xticks?
`xmarks`	Show xmarks?
`mad`	Show MAD instead of IQR?
`pch`	Points type
`se.lwd`	Lines width
`se.col`	Lines color
`...`	arguments to 'plot()'

Details

Dotchart-based plot showing medians and IQRs (or mads) for every scaled variable grouped by 'groups' factor.

Alternatives: trellis designs.

Author(s)

Alexey Shipunov

See Also

`Boxplots`

Examples

```
Trees <- trees
Trees[, 4] <- sample(letters[1:3], nrow(Trees), replace=TRUE)
Linechart(Trees[, 1:3], factor(Trees[, 4]))
```

Ls *Ls*

Description

Advanced object browser

Usage

```
Ls (pos = 1, pattern, mode = "any", type = "any", exclude = "function", sort = "name")
```

Arguments

mode	which object mode to include, "any" to include all
type	which object type to include ("type" is typically, but not always an object's class attribute), "any" to include all
exclude	exclude functions (default), "none" to include all
sort	sort by name (default), "size" to sort by size
pos	specify environment, passed to ls()
pattern	optional reguler expression, passed to ls()

Details

Based on 'ls()' but outputs data frame.

Value

Data frame with object features columns.

Author(s)

Alexey Shipunov

See Also

ls

Examples

```
data(trees)
Ls()
```

Mag *Interpreter for effect sizes*

Description

Interprets R^2-related effect sizes

Usage

```
Mag(x, squared=TRUE)
```

Arguments

x	Value
squared	Is value squared?

Details

Interpreter for R^2-related effect sizes (see example).

Author(s)

Alexey Shipunov

Examples

```
aa <- apply(cor(trees), 1:2, function(.x) Mag(.x, squared=FALSE))
aa[upper.tri(aa, diag=TRUE)] <- "-"
noquote(aa)
```

MDSv *MDS: explained variance (surrogate)*

Description

MDS: explained variance (surrogate)

Usage

```
MDSv(scores)
```

Arguments

scores	Data frame or matrix with values (e.g., result of 'isoMDS()'

Details

MDS explained variance (surrogate, regression-based)

Value

Numeric vector, one values per column of scores

Author(s)

Alexey Shipunov

Examples

```
iris.dist <- dist(iris[, 1:4], method="manhattan")
iris.dist[iris.dist == 0] <- abs(jitter(0))
library(MASS)
iris.m <- isoMDS(iris.dist)
cor(iris[, 1:4], iris.m$points) # MDS loadings surrogate
MDSv(iris.m$points) # MDS explained variance surrogate
```

Miney *Miney game*

Description

Miney minesweeper game

Usage

```
Miney(n, ucol="black", gcol="white", bcol="red")
```

Arguments

n	indicates the size of the matrix to play, i.e. n=5 results in n x n = 5 x 5 matrix.
ucol	color of unknown cells
gcol	color of good cells
bcol	color of bad cells

Details

Miney game, modified from 'miney' package of Roland Rau. See also the 'fun' package.

Author(s)

Alexey Shipunov

Examples

```
## interactive command:
Miney(3)
```

`Misclass` *Misclassification (confusion) table*

Description

Misclassification (confusion) table

Usage

```
Misclass(pred, obs)
```

Arguments

`pred`	Predicted classification
`obs`	Observed classification

Details

'Misclass()' produces misclassification (confusion) 2D table based on two classifications.

Alternatives: confusion matrix from 'caret::confusionMatrix()'

It is assumed that group orders are concerted.

Author(s)

Alexey Shipunov

Examples

```
iris.dist <- dist(iris[, 1:4], method="manhattan")
iris.hclust <- hclust(iris.dist)
iris.3 <- cutree(iris.hclust, 3)
Misclass(iris.3, iris[, 5])
```

Missing.map *Textual plot of missing data*

Description

Textual plot of missing data

Usage

```
Missing.map(df)
```

Arguments

df	Data frame with any data

Details

'Missing.map()' makes textual plot of missing data, inspired by 'DescTools::PlotMiss()'.

Author(s)

Alexey Shipunov

Examples

```
sl <- read.table("http://ashipunov.info/shipunov/open/salix_leaves.txt", h=TRUE)
Missing.map(sl)
```

MrBayes *Calls MrBayes*

Description

A slight improvement of 'ips::mrbayes()'

Usage

```
MrBayes(x, file="", nst=6, rates="invgamma", ngammacat=4, nruns=2, ngen=1e+06,
 printfreq=100, samplefreq=10, nchains=4, savebrlens="yes", temp=0.2, burnin=10,
 contype="allcompat", run=FALSE, simple=TRUE, exec="mb-mpi")
```

Arguments

x	The object to process (must be 'DNAbin' class)
file	A character string, giving the name of the MrBayes input file.
nst	An integer giving the number of rates in the model of sequence evolution.
rates	A character string; allowed are "equal", "gamma", "propinv", "invgamma", and "adgamma"; the default is "equal".
ngammacat	An integer; the number rate categories for the discretized Gamma distribution; the default is '4'.
nruns	An integer; the number of runs.
ngen	An integer; the number of states of the MCMC.
printfreq	An integer; the interval between states of the MCMC to be printed on the screen
samplefreq	An integer; the interval between states of the MCMC to be sampled.
nchains	An integer; number of Metropolis coupled MCMCs in each run.
savebrlens	Logical; shall branch lengths be saved.
temp	0.2
burnin	An integer; the number of samples from the MCMC to be discarded prior to further analysis.
contype	A character string; the type of consensus tree calculated from the posterior distribution of treeseither "halfcompat" (majority-rule consensus tree) or "allcombat" (strict consensus tree).
run	Logical; 'run = FALSE' will only print the NEXUS file, 'run = TRUE' will also start the MCMC runs, if the 'path' argument is correctly specified.
simple	New option: if TRUE (default), then outputs tree in the format readable by functions from 'ape' package
exec	New option: name of UNIX executable (to appow multithreaded version)

Details

'MrBayes()' is a slight improvement of 'ips::mrbayes()'. Please see its documentation for clarity and other options.

'MrBayes()' has two more options, it also both views and saves output (this works only on UNIX).

Author(s)

Alexey Shipunov

See Also

ips::mrbayes

Examples

```
require(ips)
data(ips.cox1)
x <- ips.cox1[, 100:140]
## Not run:
## requires MrBayes installation
MrBayes(x, file="cox1", ngen=100, run=TRUE)

## End(Not run)
```

Normality *Check normality*

Description

Check normality through Shapiro-Wilks test

Usage

```
Normality(x, p=0.05)
```

Arguments

x	numerical vector
p	level of signficance

Details

Normality via Shapiro-Wilks test. Kolmogorov-Smirnov is apparently too weak for small samples. The word of caution: this function only *helps* to decide if the data complains with parametric methods ("normal").

Value

Character vector of length one.

Author(s)

Alexey Shipunov

See Also

qqnorm, hist, rnorm

Examples

```
Normality(rnorm(100))
sapply(trees, Normality)
```

Overlap *Polygons' overlap*

Description

Polygons' overlap

Usage

```
Overlap(ppts)
Overlap2(ppts)
## S3 method for class 'Overlap'
summary(object, ...)
```

Arguments

ppts	Hulls information (e.g., output from 'Hulls()')
object	Object of the class 'Overlap'
...	Additional arguments

Details

Overlap() calculates polygons (hulls) overlap (it requires 'PBSmapping' package).

'Overlap2()' is deprecated (it requires 'gpclib' package).

Author(s)

Alexey Shipunov

See Also

Hulls

Examples

```
iris.pca <- prcomp(iris[, 1:4], scale=TRUE)
iris.p <- iris.pca$x[, 1:2]
iris.h <- Hulls(iris.p[, 1:2], as.numeric(iris[, 5]), plot=FALSE)
iris.o <- Overlap(iris.h)
summary(iris.o)
```

`pairwise.Eff` *Pairwise table of effects with magnitudes*

Description

Pairwise table of effects with magnitudes

Usage

```
pairwise.Eff(vec, fac, eff="K", dec=2)
```

Arguments

`vec`	Values
`fac`	Groups
`eff`	Effect
`dec`	Decimals to round

Details

Pairwise table of effect sizes.

At the moment, classic Lyubischev's K (a.k.a. SSSMD), 'effsize::cliff.delta()' and 'effsize::cohen.d()' supported.

Value

List with test outputs.

Author(s)

Alexey Shipunov

Examples

```
hwc <- read.table("http://ashipunov.info/data/hwc.txt", h=TRUE)
pairwise.Eff(hwc$WEIGHT, hwc$COLOR)
pairwise.Eff(hwc$WEIGHT, hwc$COLOR, eff="cohen.d")
pairwise.Eff(hwc$WEIGHT, hwc$COLOR, eff="cliff.delta")
```

`pairwise.Rro.test` *Robust rank order test post hoc derivative*

Description

Robust rank order test post hoc derivative

Usage

```
pairwise.Rro.test(x, g, p.adjust.method="BH")
```

Arguments

`x`	Values
`g`	Groups
`p.adjust.method`	See '?p.adjust'

Details

'pairwise.Rro.test()' is the Robust rank order test post hoc derivative.

Value

List with test outputs

Author(s)

Alexey Shipunov

See Also

`Rro.test`

Examples

```
pairwise.Rro.test(airquality$Ozone, airquality$Month)
```

`pairwise.Table2.test` *Pairwise Chi-squared or Fisher test for 2-dimensional tables*

Description

Pairwise Chi-squared or Fisher test for 2-dimensional tables

Usage

```
pairwise.Table2.test(tbl, names=rownames(tbl), p.adjust.method="BH", exact=FALSE, ...)
```

Arguments

`tbl`	Contingency table
`names`	Level names
`p.adjust.method`	See '?p.adjust'
`exact`	Run exact test?
`...`	Arguments to test function

Details

Pairwise Chi-squared or Fisher test for 2-dimensional tables. Alternatives: 'NCStats::chisqPostHoc()'; 'fifer::chisq.post.hoc()'.

Value

List with test outputs.

Author(s)

Alexey Shipunov

Examples

```
titanic <- margin.table(Titanic, c(1, 4))
chisq.test(titanic)
pairwise.Table2.test(titanic)
```

Peaks *Find local maxima*

Description

Find local maxima

Usage

```
Peaks(series, span=3, do.pad=TRUE)
```

Arguments

series	Numerical vector
span	Window size
do.pad	Padding

Details

Finding peaks in a simple dataset.

Author(s)

Alexey Shipunov

Examples

```
## count peaks on joint histogram, this suggests number of clusters
histdata <- hist(apply(scale(iris[, -5]), 1, function(.x) sum(abs(.x))), breaks=10, plot=FALSE)
sum(Peaks(histdata$counts)) # 3 is the first value after 1 and does not change when 8 < breaks < 22
```

Phyllotaxis *Plant phyllotaxis*

Description

Outputs the plant phyllotaxis formula or angle of divergence

Usage

```
Phyllotaxis(n, angle=FALSE)
Fibonacci(x)
```

Arguments

n	non-negative integer
angle	if TRUE, output angle of divergence
x	non-negative integer

Details

'Fibonacci(x)' calculates the n's Fibonacci's number, it is the rare case that is not exercise but really used for work.

'Phyllotaxis(n)' uses 'Fibonacci(x)' to output the phyllotaxis formula (see examples) or (if 'angle=TRUE') the angle of divergence.

Value

Number or character vector of length one.

Author(s)

Alexey Shipunov

Examples

```
sapply(1:10, Fibonacci)
sapply(1:10, Phyllotaxis)
sapply(1:10, Phyllotaxis, angle=TRUE)
```

Pleiad *Correlation circles (correlation pleiads)*

Description

Correlation circles (correlation pleiads)

Usage

```
Pleiad(tbl, abs=FALSE, corr=FALSE, dist=FALSE, treshold=FALSE,
 circ=list(1, 1, 1), breaks=5, auto=TRUE, gr=6, lwd=NULL, lty=NULL, lcol=NULL,
 abbr=-1, lbltext="internal", lblcex=1, off=1.09, hofft=0.07, hoff=1.02, legend=TRUE,
 legtext=1, legpos="topright", leghoriz=FALSE, show.int=FALSE, dig.lab=1, ...)
```

Arguments

tbl	Data: square matrix
abs	If TRUE, uses absolute values istead of real
corr	If TRUE, uses absolute values istead of real and cuts from 0 to 1, this is good for correlation matrices
dist	If TRUE, converts distance matrix to the data frame – good for "dist" objects
treshold	If this is (saying) =.5, selects for plotting (with lty=1) only those values which are >.5
circ	Line type, width and color for the cirle; if first or third =0, no cicrle
breaks	How to cut() values, if "cramer", then =c(0, .1, .3, .5, 1)
auto	If FALSE, specify lwd, lty and lcol
gr	Grayscale scheme starts from 6 breaks
lwd	If autolines=FALSE, change to vector concerted with breaks
lty	If autolines=FALSE, change to vector concerted with breaks
lcol	If autolines=FALSE, change to vector concerted with breaks; if length(lcol) == 1, all lines are of particular color
abbr	If =-1, no abbreviation; if =0, no labels; other values run abbreviate(..., abbr)
lbltext	If this is a vector starting from something else, will replace dimnames
lblcex	Magnification of labels
off	Radial offset of labels, be careful!
hofft	Treshold determining which labels are rigtmost/leftmost, hofft=0 put all labes into this group
hoff	Horizontal offset for rightmost/leftmost labels; hoff=1 removes offset
legend	If FALSE, no legend
legtext	If =1 then "weaker ... stronger"; if =2, shows cutting intervals; if =3, then 1:5; if >3, issues error
legpos	This is from 'legend()'
leghoriz	Equal to 'horiz=' from 'legend()'
show.int	Show intervals in (...] form
dig.lab	dig.lab for 'cut()'
...	Arguments to 'points()'

Details

Correlation circles (correlation pleiads). Based on the works of Petr Terentjev's (Saint Petersburg) school Alternatives: packages "igraph", "circlize" and similar It is probably a good idea to order data entries with hierarchical clustering results to optimize resulted graph

Value

Retunts data frame with position of points, helps in subsequent plot enhancing

Author(s)

Alexey Shipunov

Examples

```
l.c <- cor(datasets::longley, method="spearman", use="pairwise")
Pleiad(l.c, corr=TRUE, legtext=2, pch=21, cex=2, bg="white", breaks=3, gr=3, hoff=1, show.int=TRUE)

dr <- read.table("http://ashipunov.info/shipunov/open/drosera.txt", h=TRUE)
dr.c <- cor(dr[, -1], method="spearman", use="pairwise")
Pleiad(dr.c, corr=TRUE, legtext=2, pch=19, cex=1.2)
```

---

plot.nnet *Plots 'nnet' object*

Description

Plots 'nnet' object

Usage

```
## S3 method for class 'nnet'
plot(x, ..., nid=TRUE, all.out=TRUE, all.in=TRUE, wts.only=FALSE,
 rel.rsc=5, circle.cex=5, node.labs=TRUE, line.stag=NULL, cex.val=1, alpha.val=1,
 circle.col="lightgrey", pos.col="black", neg.col="grey")
```

Arguments

x	'nnet' model
...	arguments to 'plot()'
nid	TRUE
all.out	TRUE
all.in	TRUE
wts.only	FALSE
rel.rsc	5
circle.cex	5
node.labs	TRUE
line.stag	NULL
cex.val	1
alpha.val	1
circle.col	"lightgrey"
pos.col	"black"
neg.col	"grey"

Details

Plots 'nnet' object. Requires 'nnet' and 'scales' libraries. Based on the code from Marcus W Beck.

Author(s)

Alexey Shipunov

Examples

```
library(nnet)
iris.train <- iris[seq(1, nrow(iris), 5), ]
iris.unknown <- iris[-seq(1, nrow(iris), 5), ]
iris.nnet <- nnet(Species ~ . , data=iris.train, size=4)
iris.predicted <- predict(iris.nnet, iris.unknown[, -5], type="class")
Misclass(iris.predicted, iris.unknown[, 5])
##
oldpar <- par(mar=c(0, 0, 0, 0))
plot(iris.nnet)
par(oldpar)
```

Plot.phylocl *Plot phylogenetic tree with clades collapsed*

Description

Plot phylogenetic tree with clades collapsed into triangles or rectangles

Usage

```
Plot.phylocl(tree, cl, strict=TRUE, what="triangles", col.ed="black",
 col.td="black", col.etr="transparent", col.ttr="transparent", col.pfl="lightgrey",
 col.pbr="black", lty.p=1, lwd.p=1, col.ct="black", ct.off=0, ct.fnt=1,
 longer="0%", ...)
```

Arguments

tree	phylo object
cl	two columns classification table
strict	default TRUE: do not join all descendants
what	default "triangles", also possible to use "rectangles"
col.ed	default "black", default edge color
col.td	default black", default tips color
col.etr	default "transparent", color to suppress original edges
col.ttr	default "transparent", color to suppress original tips
col.pfl	default "lightgrey", fill color for polygons

col.pbr	default "black", border color of polygons
lty.p	default 1, line type of polygon borders
lwd.p	default 1, line width
col.ct	default "black", color of clade labels
ct.off	default 0, text offset of clade labels
ct.fnt	default 1, text font of clade labels
longer	default "0%", percent to increase xlim to fit longer clade labels
...	options to 'ape::plot.phylo()'

Details

'Plot.phylocl()' plots phylogenetic tree with clades collapsed into triangles or rectangles.

Alternative is phytools::plot.backbonePhylo() which however requires more manual work.

Some tricks used (null plotting and transparent elements), the last one is actually useful in other ways.

Intersections and other deviated cases not controlled. However, they are really easy to spot.

All parameters of polygons should be either "scalars" or vectors of the same length as clade list (minus monotypic clades), clades are in alphabetical order. To help, list of clade names is invisibly returned in the end.

Value

Returns list of clade names.

Author(s)

Alexey Shipunov

See Also

phytools::plot.backbonePhylo()

Examples

```
aa <- read.table("http://ashipunov.info/data/atmospheres.txt", h=TRUE, sep="\t",
 as.is=TRUE)
aa.d <- hclust(dist(t(aa)))
tree <- ape::unroot(ape::as.phylo(aa.d))
cl <- data.frame(planet=c("Mercury", "Venus", "Earth", "Mars", "Jupiter", "Saturn",
 "Uranus", "Neptune"),
clade=c("Earth group", "Earth group", "Earth group", "Earth group", "Close giants",
 "Close giants", "Distant giants", "Distant giants"), stringsAsFactors=FALSE)
Plot.phylocl(tree, cl, longer="5%")
```

PlotBest.dist *Plots dotchart with best base distance method*

Description

Plots dotchart with best base distance method

Usage

```
PlotBest.dist(data, distances=c("euclidean", "maximum", "manhattan",
 "canberra", "binary", "minkowski"), dim=2, plot=TRUE)
```

Arguments

data	Data frame with values
distances	Distances to use
dim	Number of dimensions
plot	Plot?

Details

Plots the "best" base distance method. Uses correlation between multidimentional scaling of distance object and PCA of data. Two dimensions are default, change it with "dim" option.

Value

Numeric vector with correlation values (equal to the number of distances involved)

Author(s)

Alexey Shipunov

Examples

```
PlotBest.dist(iris[, -5])
```

`PlotBest.hclust` *Plots dotchart with best clustering method*

Description

Plots dotchart with best clustering method

Usage

```
PlotBest.hclust(dist, clust=c("ward.D", "ward.D2", "single", "complete",
 "average", "mcquitty", "median", "centroid"), plot=TRUE)
```

Arguments

`dist`	Distance matrix
`clust`	Clustering method
`plot`	Plot?

Details

Plots the "best" hclust method. Uses cophenetic correlation.

Value

Numeric vector with correlation values (equal to the number of clusterings involved)

Author(s)

Alexey Shipunov

Examples

```
PlotBest.hclust(dist(iris[, -5], method="manhattan"))
```

`PlotBest.mdist` *Plots dotchart with best distance method, use multiple non-base distances*

Description

Plots dotchart with best distance method, use multiple non-base distances

Usage

```
PlotBest.mdist(data, distances=c("manhattan", "euclidean", "canberra", "clark",
 "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "binomial",
 "chao", "cao", "mahalanobis", "cor.pearson", "cor.spearman", "cor.kendall",
 "gower_dist", "daisy.gower", "smirnov"), dim=2, binary.only=FALSE, plot=TRUE, ...)
```

Arguments

`data`	Data frame with values
`distances`	Distances to use
`dim`	Number of dimensions
`binary.only`	Use binary only distances?
`plot`	Plot?
`...`	Arguments to 'vegdist()'

Details

Plots the "best" distance method, uses many non-base distances from diverse packages. Uses correlation between multidimentional scaling of distance object and PCA of data. Does not include "mountford" and "raup" from vegdist() as they are very special. Two dimensions are default, change it with "dim" option.

Value

Numeric vector with correlation values (equal to the number of distances involved)

Author(s)

Alexey Shipunov

See Also

PlotBest.dist

Examples

```
PlotBest.mdist(iris[, -5], scale.pca=TRUE)
mo <- read.table("http://ashipunov.info/shipunov/open/moldino.txt", h=TRUE, sep="\t",
 row.names=1)
m1 <- t((mo > 0) * 1)
PlotBest.mdist(m1, binary.only=TRUE)
```

`Ploth` *Changes the appearance of cluster dendrogram*

Description

Changes the appearance of cluster dendrogram

Usage

```
Ploth(hclust, labels=hclust[["labels"]], lab.col=1, col=1, pch.cex=1, pch="",
 bg=0, col.edges=FALSE, ...)
```

Arguments

`hclust`	Hclust object
`labels`	Labels
`lab.col`	Label colors
`col`	Colors
`pch.cex`	Scale of points
`pch`	Point types
`bg`	Points backgrounds
`col.edges`	Colorize edges?
`...`	Arguments to 'plot()'

Details

Changes the appearance of cluster dendrogram. Be sure to check plot margins! Alternatives: "dendextend" package.

Author(s)

Alexey Shipunov

Examples

```
iris.dist <- dist(iris[, 1:4], method="manhattan")
iris.hclust <- hclust(iris.dist)
Ploth(iris.hclust, col=as.numeric(iris[, 5]), pch=16, col.edges=TRUE, horiz=TRUE,
 leaflab="none")
legend("topleft", fill=1:nlevels(iris[, 5]), legend=levels(iris[, 5]))
```

`Points` *Number of cases in each location reflected in the point size*

Description

Number of cases in each location reflected in the point size

Usage

```
Points(x, y, pch=1, centers=FALSE, scale=1, cex.min=1, col=1, na.omit=TRUE, ...)
PPoints(groups, x, y, cols=as.numeric(groups), pchs=as.numeric(groups),
 na.omit.all=TRUE, ...)
```

Arguments

`x, y`	Coordinates
`pch`	Point type
`pchs`	Types of point groups
`centers`	If TRUE, show centers of each location as a pixel-size dot (pch=".")
`cex.min`	Minimal point size
`col`	Color of points
`cols`	Color of point groups
`na.omit`	If TRUE (default), skip data points with NAs
`na.omit.all`	If TRUE (default), skip data points and corresponding factor values with NAs, then make 'na.omit' for internal 'Points()' FALSE
`scale`	Scale factor for point size
`groups`	Factor defining groups
`...`	'Points()' passes other arguments to 'points()', 'PPoints()' passes other arguments to 'Points()'

Details

Frequently, more then one data point is located in one coordinate place (so called "overplotting"). How to show overplotting? One way is 'jitter()', these is also (really advanced) 'sunflowerplot()'. 'Points()' does it in its own way: number of cases in each point will be reflected in the point size. 'Points()' is a low-level graphic function, analogous to 'points()'.

'PPoints()' is the same as 'Points()' but for multiple subgroups.

To prettify plot, it is recommended to change 'scale' and optionally also 'cex.min'.

Alternative is base R 'sunflowerplot()' but it is hard to read and there is no possibility to show multiple groups in data.

Author(s)

Alexey Shipunov

See Also

jitter, sunflowerplot

Examples

```
## colors modified via palette()
plot(iris[, 1:2], type="n")
palette(rainbow(3))
PPoints(iris[, 5], iris[, 1], iris[, 2], pchs=0, scale=0.7)
palette("default")
## now with centers, colors default, pch by group, and one NA
iris[1, 1] <- NA
plot(iris[, 1:2], type="n")
PPoints(iris[, 5], iris[, 1], iris[, 2], scale=0.7, centers=TRUE)
data(iris) ## to restore default embedded object
```

R *Imitation (!) of the modern 'R' logo*

Description

Imitation (!) of the modern 'R' logo

Usage

```
R(x, y, col.e="#B8BABF", col.l="#1E63B5", cex=12)
```

Arguments

x	x coordinate of the letter
y	y coordinate of the letter
col.e	ellipse color
col.l	letter color
cex	scale, default 12

Details

Imitation (sic!) of the modern (flat) 'R' logo. Font and proportions are not exactly the same, also there is no gradient.

Author(s)

Alexey Shipunov

See Also

Ell

Examples

```
plot(1, type="n", axes=FALSE, xlab="", ylab="")
R(1.1, 0.9, cex=25)
##
plot(1:20, type="n")
for (i in 1:20) R(i, i, cex=2)
```

`Read.fasta` *Read 'FASTA' files*

Description

Simple reading of 'FASTA' files

Usage

```
Read.fasta(file)
```

Arguments

`file`	File name

Details

Simple reading of 'FASTA' files.

Value

Data frame with two columns: 'name' and 'sequence'.

Author(s)

Alexey Shipunov

Examples

```
write(file=file.path(tempdir(), "tmp.fasta"), ">some_id\nATGC")
Read.fasta(file=file.path(tempdir(), "tmp.fasta"))
```

Read.tri.nts *Read 'NTSYSpc' files*

Description

Read a lower triangular matrix

Usage

```
Read.tri.nts(file, ...)
```

Arguments

file	File to read
...	Arguments to 'scan()'

Details

Reads a lower triangular matrix which at least in my practice, typically come from 'NTSYSpc' program.

Author(s)

Alexey Shipunov

Examples

```
url.show("http://ashipunov.info/data/distances.nts")
download.file("http://ashipunov.info/data/distances.nts", "distances.nts")
distances <- Read.tri.nts("distances.nts", skip=2)
head(distances)
```

Recode *Basic multiple recoding (similar to 'SQL' left join)*

Description

Basic multiple recoding (similar to 'SQL' left join)

Usage

```
Recode(var, from, to, char=TRUE)
Recode4(var, from, to, missed="")
RecodeR(var, from, to, char=TRUE)
Recode4R(var, from, to, missed="")
```

Arguments

var	variable to recode
from	'from' column of the recoding table
to	'to' column
char	if TRUE, do not treat character vectors as factors
missed	replace missed with something, default is ""

Details

Recode Basic multiple recoding are similar to 'SQL' left join.

Inspired from Paul Johnston (Univ. of Kansas) 'recode()' function.

Alternatives are 'car::recode()', 'lessR::Recode()' and 'mgsub' package.

To understand the idea better, look on the examples.

There are four functions:

1. Recode() – base function. If starting points ("from") are the same, only the *last* rule ("from-to" pair) has an effect If rules are chained, they still work independently (i.e., chaining has no effect).

2. Recode4() – considers missing. By default, this will change non-Recode()'d entries with empty string ("")

3. RecodeR() – running recode. If starting points ("from") are the same, only the *first* rule ("from-to" pair) has an effect. Chaining is possible

4. Recode4R() – running plus considers missing. By default, this will change non-RecodeR()'ed entries with empty string ("")

Value

Recoded vector

Author(s)

Alexey Shipunov

Examples

```
phrase <- "The quick brown fox jumps over 123 lazy dogs"
var <- unlist(strsplit(phrase, split=""))
from <- letters[1:20]
to <- rev(from)
Recode.result <- paste(Recode(var, from, to), collapse="")
Recode4.result <- paste(Recode4(var, from, to, missed="-"), collapse="")
RecodeR.result <- paste(RecodeR(var, from, to), collapse="")
Recode4R.result <- paste(Recode4R(var, from, to, missed="-"), collapse="")
from.rule <- paste(from, collapse=" ")
to.rule <- paste(to, collapse=" ")
rbind(from.rule, to.rule, phrase, Recode.result, Recode4.result, RecodeR.result, Recode4R.result)
```

`Rostova.tbl` *Calculates multiple correlation matrices (via 'factor1') and stacks them together*

Description

Calculates multiple correlation matrices (via 'factor1') and stacks them together

Usage

```
Rostova.tbl(X, GROUP, ...)
```

Arguments

`X`	Data frame or matrix with values
`GROUP`	Number of grouping variable
`...`	Aguments to 'Cor.vec()'

Details

Calculates multiple correlation matrices (via GROUP) and stacks them together.

Output is suitable for PCA, distance calculations and other multivariate methods (Rostova, 1999).

Value

Data frame with correlation structure

Author(s)

Alexey Shipunov

References

Rostova N.S. 1999. The variability of correlations systems between the morphological characters. Part 1. Natural populations of Leucanthemum vulgare (Asteraceae). Botanicheskij Zhurnal. 84(11): 50–66.

See Also

`Cor.vec`

Examples

```
Trees <- trees
Trees[, 4] <- sample(letters[1:3], nrow(Trees), replace=TRUE)
Rostova.tbl(Trees, 4)
```

Rresults *Rresults shell script*

Description

Rresults shell script

Details

'Rresults' is a bash shell script which allows to gather all R input and R textual output in one text file, and (unnamed) R graphical output in another (PDF) file. If graphical output is names, it will be saved in its own file(s).

Very good for debugging and other non-interactive working with R scripts as everything is one place.

Author(s)

Alexey Shipunov

Examples

```
cat("\"Hello, world!\"", file="hello.r")

## works only if the script is properly installed
system("Rresults hello.r")
## interactive command
file.show("hello_rresults.txt")
```

Rro.test *Robust rank order test*

Description

Robust rank order test

Usage

```
Rro.test(x1, y1)
```

Arguments

x1	Fist numerical variable
y1	Second numerical variable

Details

Robust rank order test (modification of Wilcoxon test for samples with contrasting variation)

It is a variant of Fligner-Policello test.

Alternatives: 'robustrank::mod.wmw.test()'; 'npsm::fp.test()'; 'NSM3::pFligPoli()'; 'RVAideMemoire::fp.test()'; 'robustrank' package.

Value

Returns z statistic and p-value.

Author(s)

Alexey Shipunov

Examples

```
## data from help(wilcox.test)
x <- c(0.80, 0.83, 1.89, 1.04, 1.45, 1.38, 1.91, 1.64, 0.73, 1.46)
y <- c(1.15, 0.88, 0.90, 0.74, 1.21)
Rro.test(x, y)
```

Saynodynamite *Say "no" to dynamite plots!*

Description

Say "no" to dynamite plots!

Usage

```
Saynodynamite()
```

Details

'Poster' plot to emphazise the harmfullness of so called dynamite plots. See, for example, 'http://emdbolker.wikidot.com/blog

Author(s)

Alexey Shipunov

See Also

boxplot

Examples

```
Saynodynamite()
```

`Squares` *Polygons' squares*

Description

Polygons' squares

Usage

```
Squares(ppts, relative=FALSE)
```

Arguments

`ppts`	Hulls information (e.g., output from 'Hulls()')
`relative`	Calculate relative squares?

Details

Calculates polygons' squares (requires 'PBSmapping' package).

Value

Numerical vector of squares

Author(s)

Alexey Shipunov

See Also

`Hulls`

Examples

```
iris.pca <- prcomp(iris[, 1:4], scale=TRUE)
iris.p <- iris.pca$x[, 1:2]
iris.h <- Hulls(iris.p[, 1:2], as.numeric(iris[, 5]), plot=FALSE)
iris.o <- Squares(iris.h, relative=TRUE)
```

`Str` *'str()' enchanced for data frames*

Description

Enhanced 'str()': with variable numbers, row names and missing data indication

Usage

```
Str(df)
```

Arguments

`df`	Data frame

Details

'Str()' is enhanced 'str()'. If the object is a data frame, this function captures output of internal 'str()', changes it and outputs the new one. If the object is not a data frame, then output is not changed.

'Str()' (1) shows data frame structure with column indices, (2) indicates presence of NA(s) with star (*) and (3) lists first 5 row names, if they are not default.

'Str()' does not work (therefore, passes everything back to common 'str()') with data frames which have non-atomic columns (fortunately, rare case).

Alternative: 'DescTools::Str()' which uses cycles (slower!), has less features, but works with non-atomic columns.

Author(s)

Alexey Shipunov

See Also

`str`, DescTools::`Str`

Examples

```
trees1 <- trees
row.names(trees1)[1] <- "a"
trees1[1, 1] <- NA
Str(trees)
Str(trees1)
## Not run:
trees.crazy <- trees
trees.crazy[[2]] <- trees[, 2, drop=FALSE]
Str(trees.crazy) # does not work with these crazy objects

## End(Not run)
```

Table2df *Convert table to data frame saving structure*

Description

Convert table to data frame saving structure

Usage

```
Table2df(table)
```

Arguments

table	'table' object

Details

Convert contingency table into data frame and keep structure.

Value

Data frame

Author(s)

Alexey Shipunov

Examples

```
Table2df(table(iris[, 5]))
```

Tobin *Binarize (make dummy variables)*

Description

Converts vector into matrix with binary colums

Usage

```
Tobin(var, convert.names=TRUE)
```

Arguments

var	character or numerical variable
convert.names	if TRUE (default), construct new variable names, otherwise, use unique variable values as variable names

Details

'Tobin()' transforms character or numeric vector into the matrix with 0/1 (absent/present) cells.

Two approaches are in use: through '==' operation and through the conversion into factor.

First approach also constructs new names of variables whereas the second ('convert.names=FALSE') makes variable names from names of factor levels (i.e., labels).

Other alternatives: "*dumm*" packages (there are few in CRAN).

Value

Matrix with binary columns

Author(s)

Alexey Shipunov

Examples

```
(ee <- sample(letters[1:5], 10, replace=TRUE))
Tobin(ee, conv=FALSE)
Tobin(ee, conv=TRUE)
```

Toclip *Insert content to Linux X11 clipboard*

Description

Insert content to Linux X11 clipboard (uses 'xclip')

Usage

```
Toclip(x, sep="\t", row.names=FALSE, col.names=TRUE, ...)
```

Arguments

x	data frame
sep	separator, tab by default
row.names	FALSE by default
col.names	TRUE by default
...	argumants to 'write.table()'

Details

Linux-specific. Inserts data frame to Linux X11 clipboard (not primary or secondary). Useful for interface with speadsheets.

Works if 'xclip' utility is already installed.

Alternative with more flexibility: 'clipr' package.

Author(s)

Alexey Shipunov

Examples

```
## Not run:
aa <- data.frame(1:3) # Linux- (and X11-) specific
Toclip(aa) # then load the content into speadsheet

## End(Not run)
```

`Topm` *Stacks correlation matrix*

Description

Stacks (correlation) matrix and selects values which are above the “level”

Usage

```
Topm(X, level=0.45, values=0, corr=TRUE, square=TRUE)
```

Arguments

`X`	Data frame or matrix with values
`level`	Treschold
`values`	If > 0, ignores "level" and outputs until reaches number, if "all", outputs all values
`corr`	If FALSE, does not show magnitude
`square`	If FALSE, does not use lower triangle, some rows could be redundant

Details

’Topm()’ stacks (correlation) matrix and selects (and sorts) values which are above the “level”.

Good for the analysis of correlation matrices.

Value

Data frame with correlation values

Author(s)

Alexey Shipunov

See Also

`Cor`

Examples

```
Topm(cor(trees), corr=TRUE)
```

VTcoeffs *Effect sizes of association between categorical variables*

Description

Effect sizes of association between categorical variables

Usage

```
VTcoeffs(table)
```

Arguments

table	Contingency table

Details

Association between categorical variables.

Calculates Cramer's V and Tschuprow's, original and corrected (Bergsma, 2013)

Alternative: 'vcd::assocstats()'

Includes magnitude interpretation for original Cramer's V (for df < 6).

Value

Data frame with coefficients, values and tables.

Author(s)

Alexey Shipunov

Examples

```
x <- margin.table(Titanic, 1:2)
VTcoeffs(x)
VTcoeffs(x)[2, ] # most practical
```

`Write.fasta` *Write 'FASTA' files*

Description

Simple writing of 'FASTA' files

Usage

```
Write.fasta(df, file)
```

Arguments

`df`	Name of data frame
`file`	File name

Details

Simple writing of 'FASTA' files. If the data frame has more then two columns, only two first columns will be used (with warning).

Value

'FASTA' file on the disk.

Author(s)

Alexey Shipunov

Examples

```
ff <- data.frame(one="some_id", two="ATGC", three="something else")
Write.fasta(ff, file=file.path(tempdir(), "tmp.fasta")) # warning will be produced
file.show(file=file.path(tempdir(), "tmp.fasta")) # interactive
```

`Xpager` *Separate terminal pager for Linux*

Description

Separate terminal pager for Linux X11 (uses some terminal and 'less')

Usage

```
Xpager(pager="xterm")
```

Arguments

pager	name of the terminal application to use, or "old" for the default

Details

Linux pager in the new terminal window. 'xterm' is default, there is also setting for 'mate-terminal'; 'konsole' (KDE terminal) and 'gnome-terminal' are easy to add.

Run Xpager("old") to restore default behavior.

BTW, for some reason 'editor()' does not work this way.

Author(s)

Alexey Shipunov

Examples

```
## Linux- (and X11-) specific
Xpager()
?help
Xpager("old")
?help
```

Index