

Package ‘pgirmess’

December 11, 2009

Date 2009/12/11

Version 1.4.3

Title Data analysis in ecology

Author Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr>

Description Miscellaneous functions for analysis and display of ecological and spatial data

Maintainer Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr>

Suggests boot, gstat, maptools, MASS, nlme, sp, spdep, splancs

License GPL (>= 2)

URL <http://perso.orange.fr/giraudoux/>

Encoding latin1

Repository CRAN

Date/Publication 2009-12-11 07:03:56

R topics documented:

| | |
|-------------------------|----|
| CI | 2 |
| classnum | 3 |
| cormat | 4 |
| correlog | 5 |
| deg2dec | 7 |
| diag2edge | 8 |
| difshannonbio | 9 |
| dirProj | 10 |
| dirSeg | 11 |
| distNode | 12 |
| distSeg | 13 |
| distTot | 14 |
| expandpoly | 15 |

| | |
|----------------|-----------|
| friedmanmc | 16 |
| gps2gpx | 17 |
| kruskalmc | 18 |
| ks.gof | 19 |
| pairsrp | 20 |
| pave | 21 |
| pclig | 23 |
| permcont | 24 |
| PermTest | 25 |
| piankabio | 26 |
| piankabioboot | 27 |
| polycirc | 28 |
| polycirc2 | 29 |
| postxt | 30 |
| preybiom | 31 |
| print.mc | 32 |
| readVista | 32 |
| rmls | 34 |
| Segments | 34 |
| selMod | 35 |
| shannon | 37 |
| shannonbio | 38 |
| shannonbioboot | 39 |
| siegelp179 | 40 |
| tabcont2categ | 40 |
| trans2pix | 41 |
| trans2seg | 42 |
| TukeyHSDs | 43 |
| uploadGPS | 44 |
| val4symb | 45 |
| valchisq | 46 |
| write.delim | 47 |
| writeGPX | 48 |
| Index | 49 |

CI

Confidence interval of percentages

Description

Provides a $n \times 2$ matrix with the lower limit (column 1) and upper limit (column 2) of the 95 percent confidence interval of percentages

Usage

```
CI(x, y, tottrials = FALSE)
```

Arguments

| | |
|-----------------------|---|
| <code>x</code> | a vector with the number of positive observations |
| <code>y</code> | a vector of the same length as <code>x</code> with the number of negative observations, or of the total number of observations |
| <code>totrials</code> | if false (the default) <code>y</code> is the number of negative observations; if true, <code>y</code> is the total number of observations |

Details

Wrapper of `prop.test()`.

Value

A matrix of `length(x)` rows and 2 columns. Column 1: lower limit; column 2: upper limit of the 95 percent confidence interval

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr

See Also

[prop.test](#)

Examples

```
x<-c(2,10,7,8,7) # eg: number of positive cases
y<-c(56,22,7,20,5)# eg: number of negative cases
CI(x,y)

x<-c(2,10,7,8,7) # eg: number of positive cases
y<-c(4,11,7,16,10)# eg: total number of cases
CI(x,y,totrials=TRUE)
```

| | |
|-----------------------|--|
| <code>classnum</code> | <i>Gives an index vector of the class category of each value of a numerical vector</i> |
|-----------------------|--|

Description

Gives an index vector of the class category of each value of a numerical vector

Usage

```
classnum(x, breaks = "Sturges")
```

Arguments

`x` a vector of values for which the indices are desired

`breaks` one of:

- a vector giving the breakpoints between bins,
- a single number giving the number of bins,
- a character string naming an algorithm to compute the number of cells (see Details).

Details

The default for 'breaks' is "Sturges": see 'nclass.Sturges'. Other names for which algorithms are supplied are "Scott" and "FD" for "Friedman-Diaconis" (with corresponding functions 'nclass.scott' and 'nclass.FD'). Case is ignored and partial matching is used. Breaks and labels are stored as attributes.

Value

A vector of the same length as `x`, with the index of the class which each value of `x` belongs to

Author(s)

Patrick Giraudoux, pgiraudou@univ-fcomte.fr

See Also

[cut](#)

Examples

```
x<-rnorm(30)
classnum(x)
classnum(x,breaks="fd")
classnum(x, breaks=c(-1,0,1))
classnum(x,breaks=5)
```

cormat

Gives a correlation matrix and the probability of Ho for each correlation

Description

Gives a correlation matrix and the probability of Ho for each correlation estimate

Usage

```
cormat(donnees, method = "spearman", sep = FALSE)
```

Arguments

| | |
|----------------------|---|
| <code>donnees</code> | a data frame of numerics |
| <code>method</code> | a string of characters among 'pearson', 'spearman' (default), 'kendall' |
| <code>sep</code> | If true, gives the results in two matrices (default = F) |

Details

Wrapper for 'cor' and 'cor.test'. The results can be given in one or two matrices.

Value

If `sep = F` (default) a list including:

| | |
|-----------------------|---|
| <code>method</code> | The method used |
| <code>prob.cor</code> | Upper triangle, the correlations; lower triangle, the probability of Ho |

If `sep = T` a list including:

| | |
|-----------------------------|---------------------------|
| <code>method</code> | The method used |
| <code>coef.estimates</code> | The correlation matrix |
| <code>p.value</code> | The Ho probability matrix |

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[cor](#), [cor.test](#)

Examples

```
cormat(longley)
cormat(longley, sep=TRUE)
```

correlog

Computes Moran's or Geary's coefficients on distance classes

Description

Computes Moran's or Geary's coefficients on distance classes from a set of spatial coordinates and corresponding z values

Usage

```
correlog(coords, z, method="Moran", nbclass = NULL, ...)
```

Arguments

| | |
|----------------------|---|
| <code>coords</code> | a two columns array, data.frame or matrix of spatial coordinates. Column 1 = X, Column 2 = Y. |
| <code>z</code> | a vector for the values at each location. Must have the same length as the row number of coords |
| <code>method</code> | the method used. Must be "Moran" (default) or "Geary" |
| <code>nbclass</code> | number of bins. If NULL Sturges method is used to compute an optimal number |
| <code>...</code> | further arguments to pass to e.g. <code>moran.test</code> or <code>geary.test</code> |

Details

Uses the library `spdep` including `moran.test` or `geary.test`. These methods assume the data are normally distributed. Distances are euclidian and in the same unit as the spatial coordinates. Moran's Ho: I values larger than 0 due to chance; Geary's Ho: C values lesser than 1 due to chance. Correlog has `print` and `plot` methods; statistically significant values ($p < 0.05$) are plotted in red.

Value

An object of class "correlog", a matrix including:

| | |
|----------------------|------------------------|
| <code>class</code> | bin centers |
| <code>I</code> | the coefficient values |
| <code>p.value</code> | probability of Ho |
| <code>n</code> | the number of pairs |

Warning

Computing can take a long time for large data sets

Author(s)

Patrick Giraudoux pgiraud@univ-fcomte.fr and Colin Beale c.beale@macaulay.ac.uk

References

see library `spdep`

See Also

[geary.test](#), [moran.test](#)

Examples

```
library(spdep)
data(oldcol)
attach(COL.OLD)
coords<-cbind(X,Y)
res<-correlog(coords,CRIME)
plot(res)

res<-correlog(coords,CRIME,method="Geary")
plot(res)
```

deg2dec

Convert degree minutes coordinates into decimal degrees (1.60 = 2)

Description

Convert degree minutes coordinates into decimal degrees (1.60 = 2)

Usage

```
deg2dec(coord)
```

Arguments

coord Coordinates in degree minutes, minutes being separated from degrees with a dot (eg: '1.6' is equal to 2 in decimal degrees).

Value

Value(s) in decimal degrees.

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

Examples

```
deg2dec(c(1,1.3,1.6))
```

`diag2edge`*Computes the edge of a square from its diagonal*

Description

Computes the edge of a square from its diagonal.

Usage

```
diag2edge(cordseg)
```

Arguments

`cordseg` The diagonal coordinates. This can be a vector `c(x1,y1,x2,y2)`, a 2 x 2 matrix or a `data.frame` (each line a coordinate)

Details

The first point coordinates are the left top of the diagonal. The other coordinates computed are the other top of the square edge. Can be used e.g. to pass a square edge to [pave](#) in order to compute a sampling grid.

Value

A 2x2 matrix of points coordinates

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr>

See Also

[pave](#)

Examples

```
# diagonal sloping up
coord<-matrix(c(20,20,90,90),nr=2,byrow=TRUE)
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord,lty=2)
# square edge
lines(diag2edge(coord),col="red")

# diagonal sloping down
coord<-matrix(c(20,90,90,20),nr=2,byrow=TRUE)
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord,lty=2)
# square edge
```

```
lines(diag2edge(coord), col="red")

# diagonal vertical
coord<-matrix(c(20,90,20,20), nr=2, byrow=TRUE)
plot(coord, type="n", xlim=c(0,100), ylim=c(0,110), asp=1)
lines(coord, lty=2)
# square edge
lines(diag2edge(coord), col="red")
```

| | |
|---------------|---|
| difshannonbio | <i>Empirical confidence interval of the bootstrap of the difference between two Shannon indices</i> |
|---------------|---|

Description

Computes the empirical confidence interval of the bootstrap of the difference between two Shannon indices

Usage

```
difshannonbio(dat1, dat2, R = 1000, probs = c(0.025, 0.975))
```

Arguments

| | |
|-------|--|
| dat1 | a data.frame of two columns; column = category, column 2 = biomass |
| dat2 | a data.frame of two columns; column = category, column 2 = biomass |
| R | number of permutations |
| probs | the limits of the confidence interval |

Details

Designated to compare the difference between two Shannon's indices computed from two data frames. In each data frame, the first column is the category of prey item, and the second column the estimated biomass.

Value

A list with the confidence interval of H' and J'

Author(s)

patrick.giraudoux <pgiraudo@univ-fcomte.fr>

See Also

[shannonbio](#)

Examples

```

data(preylbiom)
attach(preylbiom)
jackal<-preylbiom[site=="Y" & sp=="C",5:6]
genet<-preylbiom[site=="Y" & sp=="G",5:6]

difshannonbio(jackal,genet,R=150)

```

dirProj

Computes new coordinates given bearings and distances.

Description

Computes new coordinates from bearings (North = 0) and distances

Usage

```
dirProj(df,deg=TRUE)
```

Arguments

| | |
|-----|---|
| df | a matrix or data frame of 4 columns giving x, y coordinates, bearings and distances |
| deg | if TRUE (default) bearings are in degree, otherwise in radian |

Details

Computings are based on euclidian distance. Therefore, the coordinates should be given in a projected (plan) system (e.g. UTM, Lambert, etc.) and the distance in the same units as the projection system (e.g. meters).

Value

a matrix of two columns with the projected coordinates

See Also

[distSeg](#)

Examples

```

df<-data.frame(x1=0,y1=0,alpha=runif(3,0,360),d=runif(3,0,1))
df
plot(-1:1,-1:1,type="n")
points(0,0,pch=19)
points(dirProj(df))
text(dirProj(df)[,1],dirProj(df)[,2],1:3,pos=4)

```

dirSeg *Computes segment directions.*

Description

Computes the direction of segments from the first top clockwise (North = 0)

Usage

```
dirSeg(x, deg=TRUE)
```

Arguments

| | |
|-----|---|
| x | a matrix or data frame of 4 columns giving the coordinates of each segment tops x1, y1, x2, y2 |
| deg | if TRUE (default) the output is in degrees, otherwise in radians |

Details

The first two columns give the first top coordinates, x then y, and the next two the second top coordinates.

Value

A vector of directions

See Also

[dirProj](#), [gzAzimuth](#)

Examples

```
x2<-rnorm(10)
y2<-rnorm(10)
mydata<-cbind(0,0,x2,y2)
dirs<-dirSeg(mydata)
dirs

plot(range(mydata[,c(1,3)]),range(mydata[,c(2,4)]),type="n")
Segments(mydata)
text(mydata[,3],mydata[,4],paste(round(dirs,0),"°"),cex=0.7)
```

`distNode`*Computes the distances between each nodes of a polyline.*

Description

Computes the distances between each nodes of a polyline.

Usage

```
distNode(pts, decdeg=FALSE)
```

Arguments

| | |
|---------------------|--|
| <code>pts</code> | A matrix or data.frame of the node coordinates column 1 = x, column 2 = y. |
| <code>decdeg</code> | TRUE if point coordinates are longitude-latitude decimal degrees, in which case distances are measured in meters |

Details

If `decdeg` is FALSE (default), distance computed is Euclidian. Units depends on the coordinate systems. If `decdeg = TRUE`, $D = 1852 * 60 * (180/\pi) * \arccos(\sin(la1) * \sin(la2) + \cos(la1) * \cos(la2) * \cos(\text{abs}(lg1 - lg2)))$. This method calculates the great circle distance, is based on spherical trigonometry, and assumes that:

- 1 minute of arc is 1 nautical mile
- 1 nautical mile is 1.852 km

Value

A vector of distances

See Also

[distTot](#), [distSeg](#)

Examples

```
x<-c(10, 56, 100)
y<-c(23, 32, 150)
distNode(cbind(x, y))
```

`distSeg`*Computes distances between the top coordinates of segments.*

Description

Computes the distances between the top coordinates of segments.

Usage

```
distSeg(mydata, decdeg=FALSE)
```

Arguments

| | |
|---------------------|--|
| <code>mydata</code> | A matrix or data frame of 4 columns giving the coordinates of each segment tops <code>x1, y1, x2, y2</code> |
| <code>decdeg</code> | TRUE if point coordinates are longitude-latitude decimal degrees, in which case distances are measured in meters |

Details

If `decdeg` is FALSE (default), distance computed is Euclidian. Units depends on the coordinate systems. If `decdeg = TRUE`, $D = 1852 * 60 * (180/\pi) * \cos(\sin(la1) * \sin(la2) + \cos(la1) * \cos(la2) * \cos(abs(lg1 - lg2)))$. This method calculates the great circle distance, is based on spherical trigonometry, and assumes that:

- 1 minute of arc is 1 nautical mile
- 1 nautical mile is 1.852 km

Value

A vector of distances

See Also

[distNode](#), [distTot](#)

Examples

```
x1<-rnorm(20)
y1<-rnorm(20)
x2<-rnorm(20)
y2<-rnorm(20)
mydata<-cbind(x1, y1, x2, y2)
distSeg(mydata)
```

distTot *Computes the total length of a polyline.*

Description

Computes the total length of a polyline.

Usage

```
distTot(pts, decdeg=FALSE)
```

Arguments

| | |
|--------|--|
| pts | A matrix or data.frame of the node coordinates column 1 = x, column 2 = y. |
| decdeg | TRUE if point coordinates are longitude-latitude decimal degrees, in which case distances are measured in meters |

Details

If decdeg is FALSE (default), distance computed is Euclidian. Units depends on the coordinate systems. If decdeg = TRUE, $D = 1852 * 60 * (180/\pi) * \text{acos}(\sin(\text{la1}) * \sin(\text{la2}) + \cos(\text{la1}) * \cos(\text{la2}) * \cos(\text{abs}(\text{lg1} - \text{lg2})))$. This method calculates the great circle distance, is based on spherical trigonometry, and assumes that:

- 1 minute of arc is 1 nautical mile
- 1 nautical mile is 1.852 km

Value

A numeric distance.

See Also

[, distNode](#), [distSeg](#)

Examples

```
x<-c(10, 56, 100)
y<-c(23, 32, 150)
distTot(cbind(x, y))
```

| | |
|------------|---|
| expandpoly | <i>Homothetia (size expansion) of a polygon</i> |
|------------|---|

Description

Compute the new coordinates of polygon expanded by a factor.

Usage

```
expandpoly(mypol, fact)
```

Arguments

| | |
|-------|---|
| mypol | matrix or data.frame of polygon coordinates |
| fact | expansion factor (eg 2 = 2 times, 0.5 = half, etc...) |

Value

A matrix of polygon coordinates

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[polygon](#)

Examples

```
x<-c(-5,-4.5,0,10,5)
y<-c(-10,0,5,5,-8)
poly<-cbind(x,y)
plot(-10:20,-20:10,type="n")
polygon(poly)
polygon(expandpoly(poly,1.5),border="red")
polygon(expandpoly(poly,0.5),border="blue")
```

`friedmanmc`*Multiple comparisons after Friedman test*

Description

Test of multiple comparison after Friedman test

Usage

```
friedmanmc(y, groups, blocks, probs=0.05)
```

Arguments

| | |
|---------------------|--|
| <code>y</code> | a numeric vector of data values, or a data matrix |
| <code>groups</code> | a vector giving the group for the corresponding elements of 'y' if this is a vector; ignored if 'y' is a matrix. If not a factor object, it is coerced to one. |
| <code>blocks</code> | a vector giving the block for the corresponding elements of 'y' if this is a vector; ignored if 'y' is a matrix. If not a factor object, it is coerced to one. |
| <code>probs</code> | a probability for the critical difference. |

Details

Method for formula still not implemented. Formula 7.5a (Siegel & Castellan, 1988 p 180-181) can lead to p values larger than 1 when differences between groups are small. Eventually, they are set to NA and a warning is generated.

Value

A list of class 'mc' with the following items:

| | |
|------------------------|---|
| <code>statistic</code> | statistics used |
| <code>p.value</code> | the p value of the critical difference |
| <code>dif.com</code> | a data.frame with observed and critical differences |

References

Siegel & Castellan (1988) Non parametric statistics for the behavioural sciences. Mc Graw Hill Int. Edt.

See Also

[friedman.test](#)

Examples

```

data(siegelp179)
attach(siegelp179)

friedman.test(score,treatment,block)
friedmanmc(score,treatment,block)
friedmanmc(score,treatment,block,probs=0.01)

mymatrix<-matrix(score,nc=3)
friedman.test(mymatrix)
friedmanmc(mymatrix)
detach(siegelp179)

```

 gps2gpx

Download waypoints or tracks from a GPS to a gpx file

Description

Download waypoints or tracks from a GPS to a gpx file or to the console gpx formatted

Usage

```
gps2gpx(filename="",i="garmin",f = "usb:", type = "w", invisible = TRUE)
```

Arguments

| | |
|-----------|--|
| filename | a character string naming the file to print to. If "" (the default), prints to the standard output connection |
| i | INTYPE: a supported file type, default "garmin" |
| f | INFILE: the appropriate device interface, default "usb:", on Windows for serial interfaces commonly "com4:" or similar |
| type | "w" waypoints, or "t" track, or others provided in gpsbabel |
| invisible | Under Windows, do not open an extra window |

Details

The function calls gpsbabel via the system. The gpsbabel program must be present and on the user's PATH for the function to work see <http://www.gpsbabel.org>. A .gpx suffix is added if not included in the filename. The gpx file can then be read e.g. using [readOGR](#) to a sp spatial object. Ex: `readOGR("filename.gpx", "waypoints", drop_unsupported_fields=TRUE)`, or uploaded to a GPS

See Also

[readOGR,uploadGPS](#)

Examples

```
## Not run:
gps2gpx() # download waypoints and print to the console
gps2gpx(t="t") # download tracks and print to the console
gps2gpx(filename="myfile") # download waypoints and write a gpx file

## End(Not run)
```

kruskalmc

Multiple comparison test after Kruskal-Wallis

Description

Multiple comparison test between treatments or treatments versus control after Kruskal-Wallis test

Usage

```
kruskalmc(resp, categ, probs = 0.05, cont=NULL)
```

Arguments

| | |
|-------|--|
| resp | a numeric vector of data values |
| categ | a factor object giving the group for the corresponding elements of 'x' |
| probs | a probability for the critical difference |
| cont | NULL (default) for multiple comparison between treatments; 'one-tailed' or 'two-tailed' for corresponding multiple comparisons treatments versus control; partial matching allowed |

Details

When the obtained value of a Kruskal-Wallis test is significant, it indicates that at least one of the groups is different from at least one of the others. This test helps determining which groups are different with pairwise comparisons adjusted appropriately. Those pairs of groups which have observed differences higher than a critical value are considered statistically different at the given probability (p level). Three type of multiple comparisons are implemented: comparisons between treatments, 'one-tailed' and 'two-tailed' comparison treatments versus control. The first factor level is considered the control.

For further details please consider the refence below where the method is fully described. One may also want to visit <http://pagesperso-orange.fr/giraudoux/#pgirmess>

Value

A list of class 'mc' with the following items:

| | |
|-----------|---|
| statistic | statistics used |
| p.value | the p value of the critical difference |
| dif.com | a data.frame with observed and critical differences |

References

Siegel and Castellan (1988) Non parametric statistics for the behavioural sciences. MacGraw Hill Int., New York. pp 213-214

See Also

[kruskal.test](#), to reorder factor levels see [relevel](#), [multcompLetters](#), [multcompTs](#)

Examples

```
resp<-c(0.44,0.44,0.54,0.32,0.21,0.28,0.7,0.77,0.48,0.64,0.71,0.75,0.8,0.76,0.34,0.80,0.73,0.73,0.73,0.73,0.73)
categ<-as.factor(rep(c("A","B","C"),times=1,each=6))
kruskalmc(resp, categ)
kruskalmc(resp, categ, probs=0.01)
kruskalmc(resp, categ, cont="one-tailed")
kruskalmc(resp, categ, cont="two-tailed")
```

 ks.gof

Kolmogorof-Smirnov goodness of fit test to normal distribution

Description

Kolmogorof-Smirnov goodness of fit test to normal distribution

Usage

```
ks.gof(var)
```

Arguments

| | |
|-----|------------------|
| var | a numeric vector |
|-----|------------------|

Details

A wrapper of ks.test()

Value

A list with class `"hstest"` containing the following components:

| | |
|--------------------------|--|
| <code>statistic</code> | the value of the test statistic. |
| <code>p.value</code> | a character string indicating what type of test was performed. |
| <code>alternative</code> | a character string describing the alternative hypothesis. |
| <code>method</code> | a character string indicating what type of test was performed. |
| <code>data.name</code> | a character string giving the name(s) of the data. |

Author(s)

Patrick Giraudoux patrick.giraudoux@univ-fcomte.fr

References

see `ks.test`

See Also

[ks.test](#)

Examples

```
x<-rnorm(50)
ks.gof(x)
```

`pairsrp`

Produces a matrix of scatterplot, regression coefficient and $p(H_0)$

Description

Produces a matrix with scatterplot, regression line and a loess smooth in the upper right panel; correlation coefficient (Pearson, Spearman or Kendall) and the probability of H_0 in the lower left panel

Usage

```
pairsrp(dataframe, meth = "spearman", pansmo = FALSE, abv = FALSE, lwt.cex = NULL,
```

Arguments

| | |
|------------------------|---|
| <code>dataframe</code> | a data.frame of numeric values |
| <code>meth</code> | a character string indicating which correlation coefficient is to be computed. One of 'pearson', 'kendall', or 'spearman'(default). Can be abbreviated. |
| <code>pansmo</code> | True if a loess smooth is to be plotted. Default to False. |
| <code>abv</code> | True if the variable names must be abbreviates. Default to False. |
| <code>lwt.cex</code> | character size expansion in the lower panel. |
| <code>...</code> | graphical parameters can be given as arguments to 'plot'. |

Details

This function is a wrapper for `pairs()` and `cor()`

Author(s)

Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr>

See Also

[pairs](#)

Examples

```
data(iris)
pairsrp(iris[,1:4],meth="pears",pansmo=TRUE,abv=TRUE)
```

pave

Provide square polygons or their node coordinates along a segment

Description

Provide a user-defined cellgrid of polygon squares (or square node points) along a segment. This can be used to define a sampling grid for spatial analysis.

Usage

```
pave(cordseg, yc, xc, fix.edge=NULL, ydown = TRUE, output = "list")
```

Arguments

| | |
|-----------------------|--|
| <code>cordseg</code> | the segment coordinates. This can be a vector $c(x1,y1,x2,y2)$, a 2 x 2 matrix or a data.frame (each line a coordinate) |
| <code>yc</code> | the number of segment divisions (y cells) |
| <code>xc</code> | the number of columns (x cells) |
| <code>fix.edge</code> | the edge length of a cell (user specified, default to NULL) |
| <code>ydown</code> | if TRUE (default) squares are computed decreasing y |
| <code>output</code> | a character string indicating which output is required. One of "list", "points" or "spdf". Partial match allowed |

Details

The segment must have $x1 < x2$. If not, it is automatically reordered. When "spdf" is selected the output is an object of class `SpatialPolygonsDataframe`. It has a plot method and can straightfully be handled by `writeShapePoly` (see [readShapePoly](#)) of the `maptools` library to write a shapefile. The value of the edge length of a cell can passed with the argument `fix.edge`. In this case, the coordinates of the segment right top are re-computed to adjust the cell edge to an user defined fixed value.

Value

According to the output selected, a list of polygon coordinates, a 2 column matrix with the nodes coordinates or a `SpatialPolygonsDataframe`.

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr>

See Also

[SpatialPolygonsDataFrame-class](#), [readShapePoly](#), [overlay](#), [diag2edge](#)

Examples

```
# segment sloping up
coord<-matrix(c(20,20,90,90),nr=2,byrow=TRUE)
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord)
# point grids
gr<-pave(coord,20,4,output="points") # y decreasing
points(gr)
gr<-pave(coord,20,4,output="points",ydown=FALSE) # y increasing
points(gr,col="blue")
# square polygon grids
gr<-pave(coord,20,4) # y decreasing
for (i in 1:length(gr)) polygon(gr[[i]])
gr<-pave(coord,20,4,ydown=FALSE) # y increasing
for (i in 1:length(gr)) polygon(gr[[i]],border="blue")
```

```

# segment sloping down
coord<-matrix(c(20,90,90,20),nr=2,byrow=TRUE)
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord)

# point grids
gr<-pave(coord,20,4,output="points") # y decreasing
points(gr)
gr<-pave(coord,20,4,output="points",ydown=FALSE) # y increasing
points(gr,col="blue")

# fixed edge
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord)
gr<-pave(coord,20,4,fix.edge=4,output="points")
points(gr,col="blue")

plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord)
gr<-pave(coord,20,4,fix.edge=5.5,output="points")
points(gr,col="red")

# square polygon grids
coord<-matrix(c(20,90,90,20),nr=2,byrow=TRUE)
plot(coord,type="n",xlim=c(0,100),ylim=c(0,110),asp=1)
lines(coord,lwd=2)
gr<-pave(coord,20,4)# y decreasing
for (i in 1:length(gr)) polygon(gr[[i]])
gr<-pave(coord,20,4,ydown=FALSE) # y increasing
for (i in 1:length(gr)) polygon(gr[[i]],border="blue")

## Not run:
# Writing a polygon shapefile
gr<-pave(coord,20,4,output="spdf") # y decreasing
library(maptools)
writePolyShape(gr, "myshapefilename")

## End(Not run)

```

pclig

Compute the percentage of each cell of a matrix or data.frame by row

Description

Compute the percentage of each cells of a matrix or data.frame by row

Usage

```
pclig(matr)
```

Arguments

`matr` a matrix or a data.frame

Details

Compute the percentage of each cells of a matrix by row. NA are removed.

Value

Return a matrix with percentages in each cell

See Also

[prop.table](#)

Examples

```
x<-c(2,10,7,8,7)
y<-c(56,22,7,20,5)
p<-list(cbind(x,y))
```

permcont

Random permutation of a contingency table n row x 2 columns

Description

Return a random permutation of a contingency table n rows x 2 columns keeping the marginal totals

Usage

```
permcont(Table)
```

Arguments

`Table` a contingency table

Details

The contingency table is split in a two columns table of 0/1 categories, sampled and re-organised with the function `table()`

Value

A matrix with the permuted values

Author(s)

Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr

Examples

```
tab<-cbind(n1=c(10,12,8,7,5),n2=c(4,5,8,10,12))
tab
permcont(tab)
```

 PermTest

Permutation test for lm, lme and glm (binomial and Poisson) objects

Description

Permutation test for lm, lme and glm (binomial and Poisson) objects

Usage

```
PermTest(obj, B=1000, ...)

## S3 method for class 'lm':
PermTest(obj, B=1000, ...)
## S3 method for class 'lme':
PermTest(obj, B=1000, ...)
## S3 method for class 'glm':
PermTest(obj, B=1000, ...)
```

Arguments

| | |
|-----|--|
| obj | an object of class lm, lme, or glm |
| B | number of permutations, default = 1000 |
| ... | used to pass other arguments |

Details

For glm, when the response is a two-column matrix with the columns giving the numbers of successes and failures, PermTest.glm uses permcont(); PermTest.lme requires the library nlme.

Value

A list object of class PermTest including:

| | |
|---------|----------------------------|
| p.value | the p value obtained |
| B | the number of permutations |
| call | the call |

Warning

This generic function is implemented in R language, thus can be quite slow.

Author(s)

Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr>. The implementation of PermTest.lme has been helped by Renaud Lancelot

Examples

```
library(MASS)
mylm<-lm(Postwt~Prewt,data=anorexia)
PermTest(mylm,B=250)

## Dobson (1990) Page 93: Randomized Controlled Trial :
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
glm.D93 <- glm(counts ~ outcome + treatment, family=poisson)
PermTest(glm.D93,B=250)

library(nlme)
fm2 <- lme(distance ~ age + Sex, data = Orthodont, random = ~ 1)
PermTest(fm2,B=250)
```

piankabio

Computes the Pianka's index of niche overlap

Description

Computes the Pianka's index of niche overlap

Usage

```
piankabio(dataframe1, dataframe2)
```

Arguments

dataframe1 a data frame of two columns: column 1 = dietary category, column 2 = biomass
dataframe2 a data frame of two columns: column 1 = dietary category, column 2 = biomass

Details

Computes the Pianka's index of niche overlap

Value

Return the Pianka's index

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr>

References

to do

See Also

[piankabioboot](#)

Examples

```
data (preybiom)
attach (preybiom)
jackal<-preybiom[site=="Y" & sp=="C",5:6]
genet<-preybiom[site=="Y" & sp=="G",5:6]

piankabio(jackal,genet)
```

piankabioboot

Bootstrap Pianka's index

Description

Bootstrap Pianka's index and return the limits of the empirical confidence interval specified with probs

Usage

```
piankabioboot(dataframe1, dataframe2, B = 1000, probs = c(0.025, 0.975))
```

Arguments

| | |
|------------|--|
| dataframe1 | a data frame of two columns: column 1 = dietary category, column 2 = biomass |
| dataframe2 | a data frame of two columns: column 1 = dietary category, column 2 = biomass |
| B | number of permutations |
| probs | the limits of the confidence interval |

Details

Bootstrap Pianka's index and return the limits of the empirical confidence interval specified with probs

Value

a vector of the two CI limits

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[piankabio](#)

Examples

```
data(preymbiom)
attach(preymbiom)
jackal<-preymbiom[site=="Y" & sp=="C",5:6]
genet<-preymbiom[site=="Y" & sp=="G",5:6]

piankabioboot(jackal,genet,B=100)
```

polycirc

Computes the polygon coordinates of a circle

Description

Computes the polygon coordinates of a circle

Usage

```
polycirc(radius, pts = c(0, 0), nbr = 50)
```

Arguments

| | |
|--------|---|
| radius | the length of the radius. |
| pts | the coordinates of the center. |
| nbr | the number of segments required to draw the perimeter |

Details

The matrix of coordinates can then be used with the function `polygon`

Value

A matrix of coordinates.

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr>

See Also

[polygon](#)

Examples

```
plot(1:10,1:10,type="n",asp=1)
polygon(polycirc(5),col="blue")
polygon(polycirc(2,c(5,5)),col="red")
```

polycirc2

Computes the polygon coordinates of a circle sector

Description

Computes the polygon coordinates of a circle sector

Usage

```
polycirc2(radius = 1, center = c(0, 0), edges = 50, init = pi/2, angle = pi/2)
```

Arguments

| | |
|--------|--|
| radius | the circle radius |
| center | the centre coordinates (default to x=0, y=0) |
| edges | the circular outline of the sector is approximated by a polygon with this many edges |
| init | number (in radian) specifying the starting angle |
| angle | number (in radian) specifying the sector angle |

Details

The matrix of coordinates obtained is intended to be passed to the function [polygon](#)

Value

A matrix of coordinates

See Also

[polygon](#), [polycirc](#), [floating.pie](#)

Examples

```

plot(c(-1, +1), c(-1, +1), type="n", asp=1)
polygon(polycirc2(), col="red")
polygon(polycirc2(init=pi, angle=pi/4), col="green")
polygon(polycirc2(init=1.5*pi, angle=pi/4), col="violet")
polygon(polycirc2(radius=0.5, center=c(0.5, 1)), col="blue")

polycirc2(init=pi, angle=pi/4)

```

postxt

Computes coordinates defined from their relative position on x and y in the plotting region

Description

Computes coordinates defined from their relative position on x and y in the plotting region

Usage

```
postxt(cd = "ul")
```

Arguments

`cd` a numerical vector of length 2, values comprised between 0 and 1, or one predefined among "ul", "bl", "ur", "br", "uc", "bc", "ml", "mc", "mr"

Details

The argument `cd` gives the relative position to be computed in ratio of the x or y axis. For instance `c(0.025, 0.985)` means 2.5 percents on the maximum range of the plot region on x, and 98.5 percents on y (means: close to the upper left corner of the plotting region). Predefined positions are available: "ul", upper left, "bl" bottom left, "ur" upper right, "br" bottom right, "uc" upper center, "bc" bottom center, "ml" medium left, "mc" medium center, "mr" medium right

Value

A list:

| | |
|----------------|-----------------|
| <code>x</code> | coordinate on x |
| <code>y</code> | coordinate on y |

Author(s)

Patrick Giraudoux, patrick.giraudoux@univ-fcomte.fr

See Also

[text](#)

Examples

```
plot(rnorm(30), rnorm(30), type="n")
text(postxt("ul"), "here", pos=4)
text(postxt("ur"), "here again", pos=2)
text(postxt("bc"), "again and again")
```

preybiom

Jackal and Genet diet in Algeria

Description

This data set gives the results of dietary analysis performed by Mansour Amroun in two sites of Kabylie, Algeria

Usage

```
data(preymiom)
```

Format

A data frame with 2196 observations on the following 6 variables.

faeces: a factor for faeces corresponding to faeces identification numbers

site: a factor for study sites with levels S Sebaou Y Yacouren

saison: a factor for seasons with levels H HD HP S SD SP

sp: a factor for species with levels C Jackal G Genet

category: a factor for dietary items with levels dech ind ins mammol oisauv oisdom rept
vege vegn

biomasse: a numeric vector for the weight of each dietary item

References

M. Amroun, P. Giraudoux and P. Delattre 2006 Comparative study of the diets of two sympatric carnivores - the Jackal (*Canis aureus*) and the Genet (*Genetta genetta*) - at two sites in Kabylia, Algeria. *Mammalia*, 70 (3/4): 247-254.

`print.mc` *print method for objects of class 'mc'*

Description

print method for objects of class 'mc'

Usage

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

Arguments

| | |
|------------------|--|
| <code>x</code> | an object of class 'mc' |
| <code>...</code> | further arguments to be passed to or from other methods. They are ignored in this function |

Author(s)

Patrick Giraudoux pgiraudou@univ-fcomte.fr

See Also

[kruskalmc](#), [friedmanmc](#)

Examples

```
resp<-c(0.44,0.44,0.54,0.32,0.21,0.28,0.7,0.77,0.48,0.64,0.71,0.75,0.8,0.76,0.34,0.80,0.73,0.73,0.73,0.73,0.73)
categ<-as.factor(rep(c("A","B","C"),times=1,each=6))
kruskalmc(resp, categ)
```

`readVista` *Download waypoints and tracks from a GPS*

Description

Download GPS waypoints and tracks using gpsbabel

Usage

```
readVista(i = "garmin", f = "usb:", type="w", seg=FALSE, invisible=TRUE)
```

Arguments

| | |
|------------------------|--|
| <code>i</code> | INTYPE: a supported file type, default "garmin" |
| <code>f</code> | INFILE: the appropriate device interface, default "usb:", on Windows for serial interfaces commonly "com4:" or similar |
| <code>type</code> | "w" waypoints, or "t" track, or others provided in gpsbabel |
| <code>seg</code> | track ID type: FALSE for numbers, TRUE for GPS track IDs |
| <code>invisible</code> | Under Windows, do not open an extra window |

Details

The function calls gpsbabel via the system. The gpsbabel program must be present and on the user's PATH for the function to work see <http://www.gpsbabel.org>. The function has been tested on the following Garmin GPS devices: Etrex Summit, Etrex Vista Cx and GPSmap 60CSx.

Value

A data frame of four columns:

| | |
|-----------------------|-----------------------------|
| <code>ident</code> | waypoint names or track IDs |
| <code>long</code> | longitude |
| <code>lat</code> | latitude |
| <code>altitude</code> | elevation |

Information about the data type (waypoints or tracks) and the date of download are stored as attributes.

References

<http://www.gpsbabel.org>

See Also

[readGPS](#)

Examples

```
## Not run:
mywaypoints<-readVista() # download waypoints
mytracks<-readVista(type="t") # download tracks

## End(Not run)
```

`rmls`*Select objects in the parent frame and remove them.*

Description

Select objects in the parent frame and remove them.

Usage

```
rmls ()
```

Details

This function has no arguments. This brings up a modal dialog box with a (scrollable) list of objects available in the parent frame. They can be selected by the mouse and then removed.

Author(s)

Patrick Giraudoux <patrick.giraudoux@univ-fcomte.fr>

See Also

ls, rm

Examples

```
toremove<-NULL
ls ()
  ## Not run:
  rmls()# select the object 'toremove' and click OK

## End (Not run)
ls ()
```

`Segments`*Draw line segments between pairs of points.*

Description

Draw line segments between pairs of points from a vector, matrix or data frame of 4 points coordinates x0, y0, x1, y1

Usage

```
Segments(mydata, ...)
```

Arguments

mydata a vector, matrix or data frame
 ... further graphical parameters (from 'par')

Details

a wrapper to 'segments' to handle coordinates passed as vector, matrix or data frame. Any vector is turned into a matrix of four columns.

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr>

See Also

[segments](#)

Examples

```
mydata<-cbind(rnorm(20),rnorm(20),rnorm(20),rnorm(20))
plot(range(rbind(mydata[,1],mydata[,3])),range(rbind(mydata[,2],mydata[,4])),type="n",xlab=
Segments(mydata,col=rainbow(20))

myvec<-rnorm(4)
plot(myvec[c(1,3)],myvec[c(2,4)],type="n",xlab="",ylab="")
Segments(myvec)

myvec<-rnorm(16)
plot(myvec,myvec,type="n",xlab="",ylab="")
Segments(myvec)
```

 selMod

Model selection according to information theoretic methods

Description

Handles lm, glm and list of e.g. lm, glm, nls, lme and nlme objects and provides parameters to compare models according to Anderson et al. (2001)

Usage

```
selMod(aModel, Order = "AICc", ...)

## S3 method for class 'lm':
selMod(aModel, Order = "AICc", dropNull = FALSE, selconv=TRUE, ...)
## S3 method for class 'list':
selMod(aModel, Order = "AICc", ...)
```

Arguments

| | |
|----------|--|
| aModel | a lm or glm model or a list of relevant models (see details) |
| dropNull | if TRUE, drops the simplest model (e.g. y 1) |
| Order | if set to "AICc" (default) sort the models on this parameter, otherwise "AIC" is allowed |
| selconv | if TRUE (default) keep the models for which convergence is obtained (glm object only) and with no anova singularity (lm and glm) |
| ... | other parameters to be passed as arguments (not used here) |

Details

This function provides parameters used in the information theoretic methods for model comparisons.

- .lm and glm objects can be passed directly as the upper scope of term addition (all terms added). Every model from $y \sim 1$ is computed adding one term at a time until the upper scope model is derived. This is a stepwise analysis where the terms are added sequentially and this does NOT provide all combinations of terms and interactions. Offset terms cannot be proceeded here.
- .A list of user specified lm, glm, nls, lme or nlme objects (actually any object for which AIC and logLik functions are applicable) to compare can alternately be passed.

Value

A dataframe including:

| | |
|----------|---|
| LL | the maximized log-likelihood |
| K | the number of estimated parameters |
| N2K | the number of observations/K |
| AIC | the Akaike index criterion |
| deltAIC | the difference between AIC and the lowest AIC value |
| w_i | the Akaike weights |
| deltAICc | the difference between AICc and the lowest AICc value; advised to be used when $n2K < 40$ |
| w_ic | the AICc weights |

The models examined from first to last are stored as attribute

Author(s)

Patrick Giraudoux and David Pleydell: pgiraudou@univ-fcomte.fr, dpleydel@univ-fcomte.fr

References

- Anderson, D.R., Link, W.A., Johnson, D.H. and Burnham, K.P. (2001). Suggestions for presenting the results of data analyses. *Journal of Wildlife Management*, 65, 373-378
- Burnham, K.P. and Anderson, D.R. (2002) *Model Selection and Multimodel Inference: a Practical Information-Theoretic Approach*, 2nd edn., Springer-Verlag, New York. 353 pp

See Also

[AIC,logLik](#)

Examples

```
library(MASS)
anorex.1 <- lm(Postwt ~ Prewt*Treat, data = anorexia)
selMod(anorex.1)
anorex.2 <- glm(Postwt ~ Prewt*Treat, family=gaussian,data = anorexia)
selMod(anorex.2)
anorex.3<-lm(Postwt ~ Prewt+Treat, data = anorexia)
mycomp<-selMod(list(anorex.1,anorex.2,anorex.3))
mycomp
attributes(mycomp)$models
```

shannon

Computes Shannon's and equitability indices

Description

Computes Shannon's and equitability indices

Usage

```
shannon(vect, proba=TRUE)
```

Arguments

| | |
|-------|--|
| vect | a probability vector whose sum = 1 or a frequency vector |
| proba | FALSE if the vector is frequencies and not probabilities |

Details

Computes Shannon's and equitability indices. The vector passed can be a probability vector whose sum equal 1 or a vector of frequencies (e.g. the number of food item of each category). In this case the argument proba must be set to FALSE.

Value

A vector of two values: Shannon's and equitability indices

See Also[shannonbio](#)**Examples**

```
x<-c(0.1,0.5,0.2,0.1,0.1)
sum(x)
shannon(x)
```

```
x<-rpois(10,6)
shannon(x, proba=FALSE)
```

`shannonbio`*Computes Shannon's and equitability indices from a data frame of dietary analysis (n, biomass,...)*

Description

Computes Shannon's and equitability indices from a data frame of two columns: column 1, dietary category; column 2, abundance (n, biomass,...)

Usage

```
shannonbio(data1)
```

Arguments

`data1` a data frame of two columns: column 1, dietary category; column 2, abundance (n, biomass,...)

Details

Computes Shannon's and equitability indices from a data frame of two columns: column 1, dietary category; column 2, abundance (n, biomass,...)

Value

A vector of two values: Shannon's and equitability indices

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also[shannon](#), [difshannonbio](#)

Examples

```
data (preybiom)
shannonbio (preybiom[, 5:6])
```

shannonbioboot *Bootstrap Shannon's and equitability indices*

Description

Bootstrap Shannon's and equitability indices and return an object of class boot. Confidence intervals can be computed with boot.ci().

Usage

```
shannonbioboot (data1, B = 1000)
```

Arguments

| | |
|-------|---|
| data1 | a data frame of two columns: column 1, dietary category; column 2, abundance (n, biomass,...) |
| B | number of permutations |

Details

Bootstrap Shannon's and equitability indices and return an object of class boot. Confidence intervals can be computed with boot.ci(). Requires the boot library.

Value

An object of class boot including the bootstrap statistics for H' (t1*) and J' (t2*)

Author(s)

Patrick Giraudoux <pgiraudou@univ-fcomte.fr

See Also

[boot](#), [boot.ci](#), [shannonbio](#)

Examples

```
data (preybiom)
myboot<-shannonbioboot (preybiom[, 5:6],B=100)
boot.ci (myboot, index=1,type=c ("norm", "basic", "perc")) # confidence intervals for H'
boot.ci (myboot, index=2,type=c ("norm", "basic", "perc")) # confidence intervals for J'
```

 siegelp179

Data on rats training

Description

Ranks of 18 matched groups of rats after training under three methods of reinforcement.

Usage

```
data(siegelp179)
```

Format

A data frame with 54 observations on the following 3 variables.

block Group (each of three litter mates)

treatment A factor for the type of reinforcement with levels RR RU UR

score Speed of transfer to another behaviour (the lower, the better the learning)

Details

18 blocks made of three rats of the same litter, each being given a different learning pattern (RR, RU or UR)

Source

Grosslight J.H. and Radlow R. (1956) Patterning effect of the nonreinforcement-reinforcement sequence in a discrimination situation. *Journal of Comparative and Physiological Psychology*, 49: 542-546 in Siegel & Castellan 1988. *Non parametric statistics for the behavioural sciences*. Mc Graw Hill Int. Edt.

Examples

```
data(siegelp179)
```

 tabcont2categ

Convert a contingency table (data.frame) into a presence/absence table of categories

Description

Convert a contingency table (data frame) into a data.frame of factors

Usage

```
tabcont2categ(tab)
```

Arguments

tab A data.frame (contingency table)

Details

Convert a contingency table (data frame) into a data.frame of factors

Value

A data frame

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

Examples

```
mydata<-as.data.frame(matrix(rpois(9,5),nr=3,nc=3))
names(mydata)<-LETTERS[1:3]
row.names(mydata)<-letters[1:3]

tabcont2categ(mydata)
```

trans2pix *Convert a transect coordinate file with some landmarks into a matrix with intermediate coordinates.*

Description

Convert a transect coordinate file with some landmarks and NA values in between into a matrix with intermediate coordinates.

Usage

```
trans2pix(vect)
```

Arguments

vect A two column matrix or data.frame

Details

If vect has more than two column the two first column only are read. This function computes the intermediate coordinates of each lines materialised with NA values.

Value

A matrix with the intermediate coordinates computed.

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[trans2seg](#)

Examples

```
x<-c(10,NA, NA, NA,56,NA,NA,100)
y<-c(23,NA, NA, NA,32,NA,NA,150)
cols=c("red","blue","blue","blue","red","blue","blue","red")
plot(x,y,col=cols,pch=19)
plot(trans2pix(cbind(x,y)),col=cols,pch=19)
```

trans2seg

Convert a transect coordinate file into a matrix with segment coordinates.

Description

Convert a transect coordinate file (eg: landmarks) into a matrix with segment coordinates.

Usage

```
trans2seg(vect)
```

Arguments

vect A two column matrix or data.frame

Details

The argument passed is a matrix or data.frame of two columns each row is a transect interval; each column must start (first row) and end (last row) with a landmark ; intermediate landmarks must have coordinates in the two columns of the row. Other rows must be NA values.

Value

A matrix of 4 columns to be passed eg to fonctions as "segments".

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

`trans2pix`

Examples

```
x<-c(10, NA, NA, NA, 56, NA, NA, 100)
y<-c(23, NA, NA, NA, 32, NA, NA, 150)
cols=c("red", "blue", "blue", "blue", "red", "blue", "blue", "red")
plot(x, y, col=cols, pch=19)
mysegs<-trans2seg(cbind(x, y))
segments(mysegs[,1], mysegs[,2], mysegs[,3], mysegs[,4])
```

TukeyHSDs

Simplify the list of a TukeyHSD object keeping the significant differences only.

Description

Simplify the list of a TukeyHSD object keeping the significant differences only.

Usage

```
TukeyHSDs(TukeyHSD.object)
```

Arguments

`TukeyHSD.object`
An object of calls "TukeyHSD"

Details

When TukeyHSD is used on a fitted model with large numbers of categories, the number of pairwise comparisons is extremely large ($n(n-1)/2$). TukeyHSDs simplify the TukeyHSD object keeping the significant pairwise comparisons only. A plot method exists for TukeyHSD objects.

Value

An object of class "multicomp" and "TukeyHSD"

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[TukeyHSD](#)

Examples

```
summary(fm1 <- aov(breaks ~ wool + tension, data = warpbreaks))
myobject<-TukeyHSD(fm1, "tension", ordered = TRUE)
myobject
TukeyHSDs(myobject)
```

uploadGPS

*Upload waypoints to Garmin GPS***Description**

Upload waypoints to Garmin GPS, using gpsbabel

Usage

```
uploadGPS(gpx, f = "usb:", type="w")
```

Arguments

| | |
|------|--|
| gpx | name of the .gpx file (can be created from a data frame using writeGPX) |
| f | the appropriate device interface, default "com4:", see details |
| type | 'w' for waypoints (default), 't' for track |

Details

This function uploads waypoints or a track to a garmin GPS from a '.gpx' file. gpsbabel is called via the system. Therefore gpsbabel must be installed and on the user's path, see <http://www.gpsbabel.org>. If not the default, device interface should be something as "usb:", "usb:1", or on linux "/dev/ttyUSB0", etc.

Warning

Overwrite waypoints having the same name in the GPS

See Also

[writeGPX](#)

Examples

```
## Not run:
coords<-data.frame(ID=c("C18J01", "C18J02"),Long= c(-46.996602, 47.002745),Lat=c(-6.148734,
writeGPX(coords,"mywaypoints")
uploadGPS("mywaypoint.gpx")

## End(Not run)
```

| | |
|----------|---|
| val4symb | <i>Centres a numerical vector on a parameter position and provides absolute values and colors according to negative and positive values</i> |
|----------|---|

Description

Centres a numerical vector on a parameter position and provides absolute values and colors according to negative and positive values

Usage

```
val4symb(x, FUN=mean, col = c("blue", "red"), ...)
```

Arguments

| | |
|-----|---|
| x | a numerical vector |
| FUN | a function computing a position parameter, typically mean or median . Default to mean |
| col | a character vector of 2 values, default=c("blue","red"), blue for <0, red for >=0 |
| ... | optional arguments to 'FUN' |

Value

A list with

| | |
|------|---|
| size | the absolute values of the difference to the position parameter (eg mean, median) |
| col | a character vector with 2 colors, each corresponding to positive or negative values |

Author(s)

Patrick Giraudoux, pgiraudou@univ-fcomte.fr

See Also

[symbols](#), [mean](#), [median](#), [scale](#)

Examples

```
x<-rnorm(30)
y<-rnorm(30)

z<-val4symb(rnorm(30))
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)

z<-val4symb(scale(rnorm(30)))
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)
```

```

z<-val4symb(rnorm(30),col=c("green","violet"))
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)

z<-val4symb(rnorm(30),trim=0.025)
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)

z<-val4symb(rnorm(30),median)
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)

myfun<-function(x) 20 # passes an arbitrary constant
z<-val4symb(1:30,myfun)
symbols(x,y,circle=z$size,inches=0.2,bg=z$col)

```

valchisq

Values of the partial chi-square in each cell of a contingency table

Description

Computes the values of the partial chi-square in each cell of a contingency table

Usage

```
valchisq(matr)
```

Arguments

matr a matrix (contingency table)

Details

Computes the values of the chi-square in each cell of a contingency table

Value

A matrix with the chi-square values computed

Note

No correction (e.g. Yate's etc.) is done !

See Also

valat, chisq.test

Examples

```

x <- matrix(c(12, 5, 7, 7), nc = 2)
x
valchisq(x)

```

write.delim *Write a data.frame*

Description

Write a simple data.frame into a text file with header, no row.names, fields separated by tab.

Usage

```
write.delim(x, file = "", row.names = FALSE, quote = FALSE, sep = "\t", ...)
```

Arguments

| | |
|-----------|---|
| x | a data.frame |
| file | a character string for file name |
| row.names | either a logical value indicating whether the row names of 'x' are to be written along with 'x', or a character vector of row names to be written |
| quote | a logical value or a numeric vector. If 'TRUE', any character or factor columns will be surrounded by double quotes. If a numeric vector, its elements are taken as the indices of the columns to quote. In both cases, row and column names are quoted if they are written. If 'FALSE', nothing is quoted. |
| sep | the field separator string. Values within each row of 'x' are separated by this string. |
| ... | additional arguments accepted by write.table |

Details

Simple wrapper of write.table.

Value

An ascii text file, tab delimited.

Author(s)

Patrick Giraudoux <pgiraud@univ-fcomte.fr>

See Also

[write.table](#)

Examples

```
data(preymiom)
write.delim(preymiom[1:10,]) # output to the console
write.delim(preymiom[1:10,],file="Myfile.txt") # write a file in the working directory
```

 writeGPX

Convert a data frame into a GPX file of waypoints or track

Description

Convert a data frame of labels, geographical coordinates and optionally altitude into a GPX file of waypoints or track that can be uploaded to Garmin GPS

Usage

```
writeGPX(x, filename = "", type="w")
```

Arguments

| | |
|----------|--|
| x | data.frame of three (optionally four) columns (see details) |
| filename | a character string naming the file to print to. If '''' (the default), prints to the standard output connection, the console (unless redirected by 'sink') |
| type | 'w' for waypoints (default) or 't' for track |

Details

The data frame must have three (optionally four) columns:

1. character or integer, waypoint ID for waypoints ; column not read for track
2. numeric, longitude (decimal degrees), negative for west
3. numeric, latitude (decimal degrees), negative for south
4. numeric, elevation (meters) (optional)

A suffix '.gpx' is added to the file name if not provided by user. The file obtained can be uploaded to Garmin GPS but cannot be read eg from MapSource for some reasons.

Note

for more standard GPX file, see [writeOGR](#) with arguments like layer="waypoints", driver="GPX", dataset_options="GPX_USE_EXTENSIONS=yes" can alternately be used; [readOGR](#) with arguments like layer="waypoints", drop_unsupported_fields=TRUE

See Also

[writeOGR](#)

Examples

```
coords<-data.frame(ID=c("C18J01", "C18J02"),Long= c(-46.996602, 47.002745),Lat=c(-6.148734,
writeGPX(coords) # waypoints
writeGPX(coords,type="t") # track
```

Index

*Topic **IO**

- gps2gpx, 16
- readVista, 31
- uploadGPS, 43
- writeGPX, 47

*Topic **array**

- pclig, 22
- tabcont2categ, 39
- valchisq, 45

*Topic **color**

- val4symb, 44

*Topic **connection**

- gps2gpx, 16
- readVista, 31
- uploadGPS, 43
- writeGPX, 47

*Topic **datasets**

- preybiom, 30
- siegelp179, 39

*Topic **distribution**

- permcont, 23

*Topic **dplot**

- diag2edge, 7
- pave, 20
- polycirc2, 28
- postxt, 29
- val4symb, 44

*Topic **hplot**

- pairsrp, 19
- Segments, 33

*Topic **htest**

- CI, 1
- cormat, 3
- friedmanmc, 15
- kruskalmc, 17
- ks.gof, 18
- PermTest, 24
- piankabioboot, 26
- shannonbioboot, 38

- TukeyHSDs, 42

*Topic **manip**

- deg2dec, 6
- expandpoly, 14
- polycirc, 27

*Topic **misc**

- classnum, 2
- difshannonbio, 8
- piankabio, 25
- shannon, 36
- shannonbio, 37

*Topic **models**

- selMod, 34

*Topic **print**

- print.mc, 31

*Topic **spatial**

- correlog, 4
- diag2edge, 7
- dirProj, 9
- dirSeg, 10
- distNode, 11
- distSeg, 12
- distTot, 13
- pave, 20

*Topic **utilities**

- rmls, 33
- trans2pix, 40
- trans2seg, 41
- write.delim, 46

AIC, 36

boot, 38

boot.ci, 38

CI, 1

classnum, 2

cor, 4

cor.test, 4

cormat, 3

- correlog, 4
- cut, 3
- deg2dec, 6
- diag2edge, 7, 21
- difshannonbio, 8, 37
- dirProj, 9, 10
- dirSeg, 10
- distNode, 11, 12, 13
- distSeg, 9, 11, 12, 13
- distTot, 11, 12, 13
- expandpoly, 14
- floating.pie, 28
- friedman.test, 15
- friedmanmc, 15, 31
- geary.test, 5
- gps2gpx, 16
- gzAzimuth, 10
- kruskal.test, 18
- kruskalmc, 17, 31
- ks.gof, 18
- ks.test, 19
- logLik, 36
- mean, 44
- median, 44
- moran.test, 5
- multcompLetters, 18
- multcompTs, 18
- overlay, 21
- pairs, 20
- pairsrp, 19
- pave, 7, 20
- pclig, 22
- permcont, 23
- PermTest, 24
- piankabio, 25, 27
- piankabioboot, 26, 26
- plot.correlog(*correlog*), 4
- polycirc, 27, 28
- polycirc2, 28
- polygon, 14, 28
- postxt, 29
- preybiom, 30
- print.clnum(*classnum*), 2
- print.correlog(*correlog*), 4
- print.mc, 31
- print.PermTest(*PermTest*), 24
- prop.table, 23
- prop.test, 2
- readGPS, 32
- readOGR, 16, 47
- readShapePoly, 21
- readVista, 31
- relevel, 18
- rmls, 33
- scale, 44
- Segments, 33
- segments, 34
- selMod, 34
- shannon, 36, 37
- shannonbio, 8, 37, 37, 38
- shannonbioboot, 38
- siegelp179, 39
- SpatialPolygonsDataFrame-class, 21
- symbols, 44
- tabcont2categ, 39
- text, 29
- trans2pix, 40
- trans2seg, 41, 41
- TukeyHSD, 42
- TukeyHSDs, 42
- uploadGPS, 16, 43
- val4symb, 44
- valchisq, 45
- write.delim, 46
- write.table, 46
- writeGPX, 43, 47
- writeOGR, 47