Package ‘asbio’

February 14, 2012

Type Package

Title A collection of statistical tools for biologists

Version 0.3-40

Depends R (>= 2.10), plotrix, vegan, MASS

Suggests car, odesolve, lmomco, cluster, boot, akima, tcltk

Date 2011-5-7

Author Ken Aho; many thanks to V. Winston and D. Roberts

Maintainer Ken Aho <kenaho1@gmail.com>

Description Contains functions from: ‘‘Foundational Statistics for Biologists; A Textbook Using R’’ (Under contract; CRC Press; Release: Spring 2012)

License GPL (>= 2)

LazyLoad yes

Repository CRAN

Date/Publication 2011-11-18 07:10:40

R topics documented:

`alfalfa.split.plot` *An agricultural split plot design*

Description

An experiment was conducted in Iowa in 1944 to see how different varieties of alfalfa respond to the last cutting day of the previous year (Snedecor and Cochran 1967). We know that in the fall alfalfa can either continue to grow, or stop growing and store resources in their roots for growth during the following year. Thus we might expect that later cutting dates inhibits growth for the following year. On the other hand, if plants are cut after they have gone into senescence, there should be little effect on productivity during the following year. There are two factors: 1.) variety of alfalfa (three varieties were planted in each of three randomly chosen whole plots), and 2.) the date of last cutting (Sept 1, Sept. 20, or Oct. 7). The dates were randomly chosen split plots within the whole plots. Replication was accomplished using six blocks of fields.

Usage

```
data(alfalfa.split.plot)
```

Format

The dataframe contains four variables:

yield Alfalfa yield (tons per acre).

variety Alfalfa variety. A factor with three levels "L"= Ladak, "C" = Cosack, and "R" = Ranger describing the variety of alfalfa seed used.

cut.time Time of last cutting. A factor with three levels: "None" = field not cut, "S1" = Sept 1, "S20" = Sept. 20, or "O7" = Oct. 7.

Source

Snedecor, G. W. and Cochran, G. C. (1967) *Statistical methods, 6th edition*. Iowa State University Press.

`alpha.div` *Functions for calculating alpha diversity.*

Description

The function `alpha.div` runs `Simp.index` or `SW.index` to calculate Simpson's, Inverse Simpson's or Shannon-Weiner diversities.

Alpha diversity quantifies richness and evenness within a sampling unit (replicate).

Simpson's index has a straightforward interpretation. It is the probability of reaching into a plot and simultaneously pulling out two different species. The inverse Simpson's diversity is equivalent to the probability that two randomly chosen individuals will be the same species. These measures have been attributed to Simpson (1949). While it does not allow straightforward interpretation of results, the Shannon-Weiner diversity (H') is another commonly used alpha-diversity measure based on the Kullback-Liebler information criterion (Macarthur and Macarthur 1961).

Usage

```
alpha.div(x,index)
Simp.index(x,inv)
SW.index(x)
```

Arguments

`x`	A vector or matrix of species abundances (e.g. counts). The functions assume that species are in columns and sites are in rows.
`index`	The type of alpha diversity to be computed. The function currently has three choices. `simp` = Simpson's diversity, `inv.simp`=inverse Simpson's, `shan` = Shannon-Weiner diversity.
`inv`	Logical, indicating whether or not Simpson's inverse diversity should be computed.

Value

A single diversity value is returned if x is a vector. A vector of diversities (one for each site) are returned if x is a matrix.

Author(s)

Ken Aho

References

Simpson, E. H. (1949) Measurement of diversity. *Nature*. 163: 688.

MacArthur, R. H., and MacArthur J. W. (1961) On bird species diversity. *Ecology*. 42: 594-598.

See Also

beta.div

Examples

```
library(vegan)
data(varespec)
alpha.div(varespec,"simp")
```

---

anm.ci *Animation demonstrations of confidence intervals.*

Description

Provides animated depictions of confidence intervals for μ, σ^2, the population median, and the binomial parameter p.

Usage

```
anm.ci(parent, par.val, conf = 0.95, sigma = NULL, par.type = c("mu",
"median", "sigma.sq", "p"), n.est = 100, n = 50, interval = 0.1, ...)

anm.ci.tck()
```

Arguments

parent	A parental distribution; ideally a distribution with known parameters.
par.val	True parameter value which is being estimated.
conf	Confidence level: 1-*P*(type I error).
sigma	Sigma from the normal pdf, if known.

par.type	The parameter whose confidence intervals to be estimated. There are currently four choices. These are the normal pdf parameters μ and σ^2, the population median, and the binomial parameter, p.
n.est	The number of confidence intervals to be created.
n	The sample size used for each confidence interval.
interval	The time interval for animation (in seconds). Smaller intervals speed up animation
...	Additional arguments to plot.

Details

Provides an animated plot showing confidence intervals with respect to a known parameter. Intervals which do not contain the parameter are emphasized with different colors. Loading package **tcltk** allows use of function `anm.ci.tck` which provides an interactive GUI to run the function.

Value

Returns a plot.

Author(s)

Ken Aho

See Also

Additional documentation for methods provided in: `ci.mu.t`, `ci.mu.z`, `ci.median`, `ci.sigma`, and `ci.p`.

Examples

```
## Not run:
parent<-rnorm(100000)
anm.ci(parent, par.val=0, conf =.95, sigma =1, par.type="mu")
anm.ci(parent, par.val=1, conf =.95, par.type="sigma.sq")
anm.ci(parent, par.val=0, conf =.95, par.type="median")
parent<-rbinom(100000,1,p=.65)
anm.ci(parent, par.val=0.65, conf =.95, par.type="p")
##Interactive GUI, requires package 'tcltk'
anm.ci.tck()

## End(Not run)
```

anm.coin *Animated demonstration of frequentist binomial convergence of probability using a coin flip.*

Description

Creates an animated plot showing the results from coin flips, and the resulting converence in *P*(Head) as the number of flips grows large.

Usage

```
anm.coin(flips = 1000, p.head = 0.5, interval = 0.01, show.coin = TRUE, ...)

anm.coin.tck()
```

Arguments

`flips`	The number of desired coin flips.
`p.head`	User defined probability of a head; e.g. for a fair coin `p.head` = 0.5.
`interval`	The time between animation frames, in seconds.
`show.coin`	Logical if `show.coin=TRUE` shows a second plot with coin flip results (head or tail).
`...`	Additional arguments from `link{plot}`.

Value

If `show.coin=TRUE`, returns two plots configured as a single graphical object. The first plot shows convergence in estimated *P*(Head), i.e. number of heads/number of trials, as the number of trials grows large. The second plot shows individual outcomes of coin flips. The second plot is not returned if `show.coin=TRUE` is specified. Loading **tcltk** allows use of the function `anm.coin.tck`, which in turn allows an interactive GUI to run `anm.coin`.

Author(s)

Ken Aho

See Also

`rbinom`

Examples

```
## Not run: anm.coin()
```

anm.die *Animated depiction of six-sided die throws.*

Description

Convergence in probablity for fair (or loaded) six-sided die.

Usage

```
anm.die(reps = 300, interval = 0.1, show.die = TRUE, p = c(1/6, 1/6, 1/6,
1/6, 1/6, 1/6))

anm.die.tck()
```

Arguments

reps	Number of die throws.
interval	Animation interval in frames per second.
show.die	Logical, indicating whether die outcomes should be shown
p	A vector of length six which sums to one indicating the probability of die outcomes

Author(s)

Ken Aho

See Also

anm.coin

Examples

```
## Not run:
anm.die()

## End(Not run)
```

anm.ExpDesign *Animated depiction of experimental designs*

Description

Describes random treatment allocation for twelve experimental designs.

Usage

```
anm.ExpDesign(method=c("CRD","factorial2by2","factorial2by2by2","nested",
"RCBD","RIBD","split","split.split", "SPRB","strip.split","latin","pairs"),
titles =  TRUE, interval = 0.5, iter = 30)


ExpDesign(method = c("CRD", "factorial2by2", "factorial2by2by2", "nested",
"RCBD", "RIBD", "split", "split.split", "SPRB", "strip.split", "latin",
"pairs"), titles = TRUE, ...)

anm.ExpDesign.tck()
```

Arguments

`method`	A vector listing the experimental methods to be demonstrated. Any subset of the character string `c("CRD","factorial2by2","factorial2by2by2", "nested","RCBD","RIBD","spl` `"strip.split","latin","pairs")`.
`titles`	A logical argument specifying whether or not plots should have `main` titles.
`interval`	Time length spent on each frame in animation (in seconds).
`iter`	Number of random iterations in animation.
`...`	Additional arguments from `plot`.

Details

The function returns a plot or series of plots illustrating the workings of experimental designs. Random apportionment of treatments of experimental units (EUs) is illustrated for each of twelve experimental designs. Choices are: `"CRD"`: a one-way completely randomized design,`"factorial2by2"`: a 2 x 2 design with four EUs, `"factorial2by2by2"`: a 2 x 2 x 2 factorial designs with 8 EUS, `"nested"`: a nested design with two levels of nesting,`"RCBD"` a randomized complete block design with two blocks, two treatments and four EUs,`"RIBD"`: a randomized incomplete block design with three blocks, three treatments, and six EUs,`"split"`: a split plot design with a whole plot (factor A) and a split plot (factor B),`"split.split"`: a split split-split plot design, `"SPRB"`: split plots in randomized blocks,`"strip.split"`: strip-split plot design,`"latin"`: a Latin squares design with r = 3, and `"pairs"`: a matched pairs design. The function `anm.ExpDesign.tck` provides an interactive GUI.

Author(s)

Ken Aho

See Also

samp.design

Examples

```
ExpDesign()
## Not run: anm.ExpDesign()
```

---

anm.geo.growth *Animated depictions of population growth*

Description

Animated depictions of geometric, exponential, and logistic growth.

Usage

```
anm.geo.growth(n0, lambda, time = seq(0, 20), ylab = "Abundance",
xlab = "Time", interval = 0.1, ...)

anm.exp.growth(n, rmax, time = seq(0, 20), ylab = "Abundance",
xlab = "Time", interval = 0.1, ...)

anm.log.growth(n, rmax, K, time = seq(0, 60), ylab = "Abundance",
xlab = "Time", interval = 0.1, ...)

anm.geo.growth.tck()

anm.exp.growth.tck()

anm.log.growth.tck()
```

Arguments

n0	Population size at time zero for geometric population growth.
lambda	Geometric growth rate.
time	A time sequence, i.e. a vector of integers which must include 0.
ylab	Y-axis label.
xlab	X-axis label
interval	Animation interval in seconds per frame.

`...`	Additional arguments to `plot`
`n`	Initial population numbers for exponential and logistic growth
`rmax`	The maximum intrinsic rate of increase
`K`	The carrying capacity

Details

Presented here are three famous population growth models from ecology. Geometric, exponential and logistic growth. The first two model growth in the presence of unlimited resources. Geometric growth assumes non-overlapping generations, and is computed as:

$$N_t = N_0 \lambda^t,$$

where N_t is the number of individuals at time *y*, λ is the geometric growth rate, and *t* is time.

Exponential growth allows simultaneous existience of multiple generations, and is computed as:

$$\frac{dN}{dt} = r_{max} N,$$

where r_{max} is the maximum intrinsic rate of increase, i.e. max(birth rate - death rate), and *N* is the population size. With logistic growth exponential growth is slowed as *N* approaches the carrying capacity. It is computed as:

$$\frac{dN}{dt} = r_{max} N \frac{K - N}{K},$$

where r_{max} is the maximum rate of intrinsic increase, N is the population size, and K is the carrying capacity

Installation of package **tcltk** allows implementation of all three models using GUIs.

Author(s)

Ken Aho

See Also

`anm.LVexp`, `anm.LVcomp`

Examples

```
## Not run:
anm.geo.growth(10,2.4)

## End(Not run)
```

---

anm.ls *Animated plot of least squares function.*

Description

Depicts the process of least squares estimation by plotting the least squares function with respect to a vector of estimate possibilities.

Usage

```
anm.ls(X, poss=NULL, parameter = "mu", est.lty = 2, est.col = 2,
conv=diff(range(X))/50, anim=TRUE, plot.lsfunc = TRUE, plot.res = TRUE,
interval=0.01, xlab=expression(paste("Estimates for E(X)")),...)

anm.ls.tck()
```

Arguments

`X`	A numeric vector containing sample data.
`poss`	An ordered numeric sequence of possible parameter estimates. Inclusion of the least squares estimate in the vector (e.g. $barX$ for mu will cause the least squares function be minimized at this value.
`parameter`	Parameter to be estimated. Only estimation for $E(X)$ is currently implemented. Note that if $X \sim N(\mu,\sigma^2)$ then $E(X) = \mu$.
`est.lty`	Line type to be used to indicate the least squares estimate.
`est.col`	Line color to be used to indicate the least squares estimate.
`conv`	Precision of LS function. Decreasing `conv` increases the smoothness and precision of the function.
`anim`	A logical command indicating whether animation should be used in plots.
`plot.lsfunc`	A logical command indicating whether the least-squares function should be plotted.
`plot.res`	A logical command indicating whether a plot of residuals should be created.
`interval`	Speed of animation (in frames per second). A smaller interval decreases speed. May not work in all systems; see `Sys.sleep`.
`xlab`	X-axis label.
`...`	Additional arguments to `plot`

Value

A plot of the least squares function is returned along with the least squares estimate for E(X) given a set of possibilities. The function `anm.ls.tck` provides a GUI to run the function.

Author(s)

Ken Aho

See Also

loglik.plot

Examples

```
## Not run: X<-c(11.2,10.8,9.0,12.4,12.1,10.3,10.4,10.6,9.3,11.8)
anm.ls(X)
## End(Not run)
```

---

anm.LV — *Animated depictions of Lotka-Volterra competition and exploitation models*

Description

Creates animated plots of two famous abundance models from ecology; the Lotka-Volterra competition and exploitation models

Usage

```
anm.LVcomp(n1, n2, r1, r2, K1, K2, a2.1, a1.2, time = seq(0, 200), ylab =
"Abundance", xlab = "Time", interval = 0.1, ...)

anm.LVexp(nh, np, rh, con, p, d.p, time = seq(0, 200), ylab = "Abundance",
xlab = "Time", interval = 0.1, circle = FALSE, ...)

anm.LVc.tck()

anm.LVe.tck()
```

Arguments

n1	Initial abundance values for species one. To be used in the competition function anm.LVcomp, i.e. N_1 in the competition equations below.
n2	Initial abundance values for species two in the competition function, i.e. N_2 in the competition equations below.
r1	Maximum intrinsic rate of increase for species one, i.e. r_{max1}.
r2	Maximum intrinsic rate of increase for species two in the competition model anm.LVcomp, i.e. r_{max2}.
K1	Carrying capacity for species one, i.e. K_1.
K2	Carrying capacity for species two, i.e. K_2 .
a2.1	The interspecific effect of species one on species two, i.e. the term α_{21}.

`a1.2`	The interspecific effect of species two on species one, i.e. the term α_{12}.
`nh`	Initial abundance values for the host (prey) species. To be used in the the exploitation model `anm.LVexp`, i.e. the term n_h.
`np`	Initial abundance values for the predator species in the the exploitation model, i.e. the term h_p.
`rh`	The intrinsic rate of increase for the host (prey) species, i.e. the term r_h.
`con`	The conversion rate of prey to predator, i.e. the term c.
`p`	The predation rate, i.e. the term p.
`d.p`	The death rate of predators, i.e. the term d_p.
`time`	A time sequence for which competition or exploitation is to be evaluated.
`ylab`	*Y*-axis label.
`xlab`	*X*-axis label.
`interval`	Animation speed per frame (in seconds).
`circle`	Logical, if `TRUE` a circular representation of the relation of prey and predator numbers is drawn.
`...`	Additional arguments from `plot`.

Details

The Lotka-Volterra competition and exploitation models require simultaneous solutions for two differential equations. These are solved using the function `rk4` from `odesolve`.

The interspecific competition model is based on:

$$\frac{dN_1}{dt} = r_{max1}N_1\frac{K_1 - N_1 - \alpha_{12}}{K_1},$$

$$\frac{dN_2}{dt} = r_{max2}N_2\frac{K_2 - N_2 - \alpha_{21}}{K_2},$$

where N_1 is the number of individuals from species one, K_1 is the carrying capacity for species one, r_{max1} is the maximum intrinsic rate of increase of species one, and α_{12} is the interspecific competitive effect of species two on species one.

The exploitation model is based on:

$$\frac{dN_h}{dt} = r_hN_h - pN_hN_p,$$

$$\frac{dN_p}{dt} = cpN_hN_p - d_pN_p,$$

where N_h is the number of individuals from the host (prey) species, N_p is the number of individuals from the predator species, r_h is the intrinsic rate of increase for the host (prey) species, p is the rate of predation, c is a conversion factor which describes the rate at which prey are converted to new predators, and d_p is the death rate of the predators.

The term r_hN_h describes exponential growth for the host (prey) species. This will be opposed by deaths due to predation, i.e. the term pN_hN_p. The term cpN_hN_p is the rate at which predators destroy prey. This in turn will be opposed by d_pN_p, i.e. predator deaths. Loading package **tcltk** allows one to run the GUIs in `anm.LVe.tck` and `anm.LVc.tck`.

Value

The functions return descriptive animated plots

Author(s)

Ken Aho, based on a concept elucidated by M. Crawley

References

Molles, M. C. (2010) *Ecology, Concepts and Applications, 5th edition*. McGraw Hill.

Crawley, M. J. (2007) *The R book*. Wiley

Examples

```
## Not run:

#---------------------- Competition ---------------------#
##Species 2 drives species 1 to extinction
anm.LVcomp(n1=150,n2=50,r1=.7,r2=.8,K1=200,K2=1000,a2.1=.5,a1.2=.7,time=seq(0,200))
##Species coexist with numbers below carrying capacities
anm.LVcomp(n1=150,n2=50,r1=.7,r2=.8,K1=750,K2=1000,a2.1=.5,a1.2=.7,time=seq(0,200))

#----------------------Exploitation----------------------#
#Fast cycles
anm.LVexp(nh=300,np=50,rh=.7,con=.4,p=.006,d.p=.2,time=seq(0,200))
## End(Not run)
```

`anm.samp.design` *Animated demonstration of randomized sampling designs*

Description

Animated Comparisons of outcomes from simple random sampling, stratified random sampling and cluster sampling.

Usage

```
anm.samp.design(n=20, interval = 0.5 ,iter = 30)

samp.design(n = 20)

anm.samp.design.tck()
```

Arguments

`n`	The number of samples to be randomly selected from a population of 400.
`interval`	Time length spent on each frame in animation (in seconds).
`iter`	Number of random iterations in animation.

Details

Returns a plot comparing outcomes of random sampling, stratified random sampling and cluster sampling from a population of size 400. For stratified random sampling the population is subdivided into four equally strata of size 100. and n/4 samples are taken within each strata. For cluster sampling the population is subdivided into four equally sized clusters and a census is taken from two clusters (regardless of specification of n). The function `anm.samp.design` depicts random sampling using animation

Value

A plot is returned with four subplots. (a) shows the population before sampling, (b) shows simple random sampling, (c) shows stratified random sampling, (d) shows cluster sampling. The function `anm.samp.design.tck` provides interaction with a **tcltk** GUI.

Author(s)

Ken Aho

Examples

```
samp.design(20)

#Animated demonstration
## Not run: anm.samp.design(20)
```

`AP.test` *Agresti-Pendergrast test*

Description

Provides a more powerful alternative to Friedman's test for blocked (dependent) data with a single replicate.

Usage

```
AP.test(Y)
```

Arguments

`Y`	A matrix with treatments in columns and blocks (e.g. subjects) in rows.

Details

The Agresti-Pendergrast test is more powerful than Friedman's test, given normality, and remains powerful in heavier tailed distributions (Wilcox 2005).

Value

Returns a dataframe showing the numerator and denominator degrees of freedom, F test statistic, and p-value.

Note

code based on Wilcox (2005).

Author(s)

Ken Aho

References

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

friedman.test, MS.test

Examples

```
temp<-c(2.58,2.63,2.62,2.85,3.01,2.7,2.83,3.15,3.43,3.47,2.78,2.71,3.02,3.14,3.35,
2.36,2.49,2.58,2.86,3.1,2.67,2.96,3.08,3.32,3.41,2.43,2.5,2.85,3.06,3.07)
Y<- matrix(nrow=6,ncol=5,data=temp,byrow=TRUE)
AP.test(Y)
```

---

asthma *Asthma repeated measures dataset from Littell et al. (2002)*

Description

This dataset was used by Littell (2002) to demonstrate repeated measures analyses. The effect of two asthma drugs and a placebo were measured on 24 asthmatic patients. Each patient was randomly given each drug using an approach to minimize carry-over effects (Section 15.6) Forced expiratory volume (FEV1), the volume of air that can be expired after taking a deep breath in one second, was measured. FEV1 was measured hourly for eight hours following application of the drug. A baseline measure of FEV1 was also taken 11 hours before application of the treatment.

Usage

```
data(asthma)
```

Format

The dataframe has 11 columns:

PATIENT The subjects (there were 24 patients).

BASEFEV1 A numerical variable; the baseline forced expiratory volume.

FEV11H Forced expiratory volume at 11 hours.

FEV12H Forced expiratory volume at 12 hours.

FEV13H Forced expiratory volume at 13 hours.

FEV14H Forced expiratory volume at 14 hours.

FEV15H Forced expiratory volume at 15 hours.

FEV16H Forced expiratory volume at 16 hours.

FEV17H Forced expiratory volume at 17 hours.

FEV18H Forced expiratory volume at 18 hours.

DRUG A factor with three levels `"a"` = a standard drug treatment, `"c"` = the drug under development, and `"p"` = a placebo.

Source

Littell, R. C., Stroup, W. W., and R. J. Fruend (2002) *SAS for linear models*. John Wiley and Associates.

Bayes.disc — *Bayesian graphical summaries for discrete data.*

Description

An exceedingly simple function for for summarizing a Bayesian analysis given discrete data and priors.

Usage

```
Bayes.disc(Data, Prior, data.name = "data", plot = TRUE,
c.data = seq(1, length(Prior)), ...)

Bayes.disc.tck()
```

Arguments

`Data`	A vector of data probabilities. This must be in the same order as `Prior`, i.e. if θ_1 is the first element in `Prior`, then $data|\theta_1$ must be the first element in `Data`.
`Prior`	A vector of prior probabilities, or weights.
`data.name`	A name for data in conditional statements.
`plot`	Logical, indicating whether a plot should be made.
`c.data`	A character string of names for discrete classes
`...`	Additional arguements to `plot`.

Author(s)

Ken Aho

`BDM.test` *Brunner-Dette-Munk test*

Description

One and two way heteroscedastic rank-based permutation tests. Two way designs are assumed to be factorial, i.e. interactions are tested.

Usage

```
BDM.test(Y, X)

BDM.2way(Y, X1, X2)
```

Arguments

`Y`	Vector of response data. A quantitative vector
`X`	A vector of factor levels for a one-way analysis. To be used with `BDM.test`
`X1`	A vector of factor levels for the first factor in a two-way factorial design. To be used with `BDM.2way`.
`X2`	A vector of factor levels for the second factor in a two-way factorial design. To be used with `BDM.2way`.

Details

A problem with the Kruskal-Wallis test is that, while it does not assume normality for groups, it still assumes homoscedasticity (i.e. the groups have the same distributional shape). As a solution Brunner et al. (1997) proposed a heteroscedastic version of the Kruskal-Wallis test which utilizes the F-distribution. Along with being robust to non-normality and heteroscedasticity, calculations of exact p-values using the Brunner-Dette-Munk method are not made more complex by tied values. This is another obvious advantage over the traditional Kruskal-Wallis approach.

Value

Returns a list with two components

`Q`	The "relative effects" for each group.
`Table`	An ANOVA type table with hypothesis test results.

Note

Code based on Wilcox (2005)

Author(s)

Ken Aho

References

Brunner, E., Dette, H., and A. Munk (1997) Box-type approximations in nonparametric factorial designs. *Journal of the American Statistical Association*. 92: 1494-1502.

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`kruskal.test`, `trim.test`

Examples

```
rye<-c(50,49.8,52.3,44.5,62.3,74.8,72.5,80.2,47.6,39.5,47.7,50.7)
nutrient<-factor(c(rep(1,4),rep(2,4),rep(3,4)))
BDM.test(Y=rye,X=nutrient)
```

`beta.div` *Beta diversity functions*

Description

Several methods for beta diversity calculation: Whittaker, average dissimilarity, i.e. $\bar{D}$, and half-change.

Usage

```
W.beta.div(x)
beta.dbar(x,index="steinhaus")
beta.dbar.halfchange(x,index="steinhaus")
beta.div(x,method=c("whittaker","dbar","half.change"),index="steinhaus")
```

Arguments

`x`	A community matrix with sites in rows and species in columns.
`method`	One of three possible choices in beta.div: `"whittaker"`,`"dbar"`,or `"half.change"`.
`index`	A dissimilarity or distance measure. Any measure from `get.dist` is acceptable.

Details

Beta diversity can unfortunately mean two related but different things. First, it can refer to the relationship between the total number of species in landscape, (i.e.gamma-diversity), and the richness in individual plots. As a result, this type of beta-diversity attempts to quantify community heterogeneity without consideration of spatial distance or an environmental gradient. Second, beta diversity can refer to a measure of the rate of change in species composition across a pre-defined spatial or environmental gradient (Velland 2001). The second, more complex type of measure is often called species turnover.

A simple formula for the first (non-gradient) type of beta-diversity is attributed to Whittaker (1960). It is:

$$\beta_W = \frac{\gamma}{\kappa} - 1,$$

where γ is the total number of species in the landscape, i.e gamma diversity, and κ is average plot richness. β_W attains it maximum value when no species are shared among plots. It reaches its minimum value, 0, when species composition is identical among plots. Multivariate resemblance measures are also conventionally used for measuring non-gradient beta diversity (e.g. Aho et al. 2008). The simplest approach is to find the mean dissimilarity in a distance matrix, $\bar{D}$. This approach addresses several of the problems implicit with Whittaker's function. Specifically, that Whittaker's method is 1) strongly affected by rare species and 2) considers only presence/absence, not abundance. A recommended adjustment to the distance matrix mean has been made by McCune and Grace (2002):

$$\beta_{HC} = \frac{ln(1 - \bar{D})}{ln(0.5)},$$

where $\bar{D}$ is the average dissimilarity. In this approach average distance is converted to a half change scale.

To measure the second type of beta-diversity (i.e. species turnover along a gradient) axes from Detrended Correspondence Analysis (DCA) have been used (Jongman et al. 1995). This approach is suitable because DCA utilizes Hill scaling to create its (indirect gradient) ordination axes. Complete turnover for a site in a species space occurs at a distance of about four axis units (four DCA standard deviations). DCA expands and contracts axes so that the rate of species turnover is more or less constant. Oksanen and Tonteri (1995) and others have contested this use of DCA axes and proposed more direct estimation of turnover along known (not indirect) ordination gradients.

Value

Returns beta diveristy.

Author(s)

Ken Aho

References

Aho, K., Roberts, D. W., and T. Weaver. (2008) Using geometric and non-geometric internal evaluators to compare eight vegetation classification methods. *Journal of Vegetation Science*. 19: 549-562.

Oksanen, J. and T. Tonteri. (1995) Rate of compositional turnover along gradients and total gradient length. *Journal of Vegetation Science* 6: 815-824.

Jongman, R. H. G., ter Braak, C. F. G., and O. F. R. van Tongern. (1995) *Data analysis In Community and landscape ecology*. Cambridge University Press. Cambridge, UK.

See Also

get.dist, alpha.div

Examples

```
library(vegan)
data(varespec)
W.beta.div(varespec)
```

---

bombus — *Bombus pollen data.*

Description

To investigate how pollen removal varied with reproductive caste in bemblebees (*Bombus* sp.) Harder and Thompson (1989) recorded the proportion of pollen removed by thirty five bumblebee queens and twelve worker bees.

Usage

```
data(bombus)
```

Format

This data frame contains the following columns:

pollen A numeric vector indicating the proportion of pollen removed.

caste A character string vector indicating whether a bee was a worker "W" or a queen "Q".

Source

Harder, L. D. and Thompson, J. D. 1989. Evolutionary options for maximizing pollen dispersal in animal pollinated plants. *American Naturalist* 133: 323-344.

boot.ci.M *Bootstrap CI of M-estimators differences of two samples*

Description

Creates a bootstrap confidence interval for location differences for two samples. The default location estimator is the Huber one-step estimator, although any estimator can be used. The function is similar to a function written by Wilcox (2005) but does not compute two-tailed p-values as these are often in conflict with the calculated confidence intervals. The default method for bootstrap confidence intervals is the percentile method which is recommended by Wilcox (2005) for M-estimators.

Usage

```
boot.ci.M(X1, X2, alpha = 0.05, est = huber.one.step, R = 1000,
type = "perc")
```

Arguments

X1	Sample from population one.
X2	Sample from population two.
alpha	Significance level.
est	Location estimator; default is the Huber one step estimator.
R	Number of bootstrap samples.
type	Method for computing bootstrap confidence intervals. Other alternatives are given in boot.ci from the library boot.

Value

Returns a list with one component, a dataframe containing summary information from the analysis:

R.used	The number of bootstrap samples used. This may not = R if NAs occur because $MAD = 0$.
ci.type	The method used to construct the confidence interval.
conf	The level of confidence used.
se	The bootstrap distribution of differences standard error.
original	The original, observed difference.
lower	Lower confidence bound.
upper	Upper confidence bound.

Author(s)

Ken Aho

References

Manly, B. F. J. (1997) *Randomization and Monte Carlo methods in biology, 2nd edition*. Chapman and Hall, London.

See Also

bootstrap, boot

Examples

```
## Not run:
X1<-rnorm(100,2,2.5)
X2<-rnorm(100,3,3)
boot.ci.M(X1,X2)

## End(Not run)
```

---

bootstrap — *A simple function for bootstrapping*

Description

The function serves as a simplified alternative to the function boot from the library boot.

Usage

```
bootstrap(data, statistic, R, matrix = FALSE)
```

Arguments

data	Raw data to be bootstrapped. A vector or quantitative data or a matrix if matrix =TRUE.
statistic	A function whose output is a statistic (e.g. a sample mean). The function must have only one argument, a call to data.
R	The number of bootstrap iterations.
matrix	A logical statement. If matrix = TRUE then rows in the matrix are sampled as multivariate observations.

Details

With bootstrapping we sample with replacement from a dataset of size *n* with n samples R times. At each of the R iterations a statistical summary can be created resulting in a bootstrap distribution of statistics.

Value

Returns a bootstrap distribution of a statistic.

Author(s)

Ken Aho

References

Manly, B. F. J. (1997) *Randomization and Monte Carlo methods in biology, 2nd edition*. Chapman and Hall, London.

See Also

boot

Examples

```
library(vegan)
data(varespec)
# A partial set of observations from a single plot for a Scandinavian
# moss/vascular plant/lichen survey.
site18<-data.frame(t(varespec[1,][1:20]))

#Shannon-Weiner diversity
SW<-function(data){
d<-data[data!=0]
p<-d/sum(d)
-1*sum(p*log(p))
}

bootstrap(site18[,1],SW,R=1000,matrix=FALSE)
```

---

bound.angle *Angle of azimuth to a boundary.*

Description

The function calculates the angle of azimuth from a Cartesian coordination given in X and Y to a nearest neighbor coordinate given by nX and nY. The nearest neighbor coordinates can be obtained using the function near.bound.

Usage

```
bound.angle(X, Y, nX, nY)
```

Arguments

X	Cartesian *X* coordinate of interest.
Y	Cartesian *Y* coordinate of interest.
nX	Cartesian *X* coordinate of nearest neighbor point on a boundary.
nY	Cartesian *Y* coordinate of nearest neighbor point on a boundary.

Details

The function returns the nearest neighbor angles (in degrees) with respect to a four coordinate system ala ARC-GIS Near(Analysis). Output angles range from $-180°$ to $180°$: $0°$ to the East, $90°$ to the North, $180°$ (or $-180°$) to the West, and $-90°$ to the South.

Value

Returns a vector of nearest neighbor angles.

Author(s)

Ken Aho

See Also

near.bound, prp

Examples

```
bX<-seq(0,49)/46
bY<-c(4.89000,4.88200,4.87400,4.87300,4.88000,4.87900,4.87900,4.90100,4.90800,
4.91000,4.93300,4.94000,4.91100,4.90000,4.91700,4.93000,4.93500,4.93700,
4.93300,4.94500,4.95900,4.95400,4.95100,4.95800,4.95810,4.95811,4.95810,
4.96100,4.96200,4.96300,4.96500,4.96500,4.96600,4.96700,4.96540,4.96400,
4.97600,4.97900,4.98000,4.98000,4.98100,4.97900,4.98000,4.97800,4.97600,
4.97700,4.97400,4.97300,4.97100,4.97000)

X<-c(0.004166667,0.108333333,0.316666667,0.525000000,0.483333333,0.608333333,
0.662500000,0.683333333,0.900000000,1.070833333)
Y<-c(4.67,4.25,4.26,4.50,4.90,4.10,4.70,4.40,4.20,4.30)
nn<-near.bound(X,Y,bX,bY)

bound.angle(X,Y,nn[,1],nn[,2])
```

`bplot` *Barplots with standard errors and CIs for pairwise comparisons.*

Description

Creates barplots displaying treatment measures of location (e.g. means, or medians) along with error bars (i.e. standard errors or confidence intervals or IQRs). Can also display letters indicating if results were significant after adjustment for simultaneous inference.

Usage

```
bplot(y, x, int = "CI", conf = 0.95, plot.ci = TRUE,
bar = TRUE, simlett = FALSE, bar.col = "gray", lett = NULL, exp.fact = 2,
xlab = "x", ylab = "y", err = "y", sfrac = 0.01, gap = 0, slty = par("lty"),
scol = NULL,  pt.bg = par("bg"),order=FALSE,names.arg=NULL, width=1,
loc.meas=mean, cex.lab=1, cex.axis=1, cex.names=1, las=par("las"),
horiz = FALSE, ylim = NULL,...)
```

Arguments

`y`	A quantitative vector representing the response variable.
`x`	A categorical vector representing treatments (e.g. factor levels).
`int`	Type of error bar to be drawn, one of `"SE"`, `"CI"`, `IQR`, or `"IQR.CI"`. IQR-derived confidence intervals are based on +/-1.58 IQR/sqrt(n) and provide an approximate 95% confidence interval for the difference in two medians. The measure can be attributed to Chambers et al. (1983, p. 62), given McGill et al. (1978, p. 16). It is based on asymptotic normality of the median and assumes roughly equal sample sizes for the two medians being compared. The interval is apparently insensitive to the underlying distributions of the samples.
`conf`	Level of confidence, 1 - *P*(type I error).
`plot.ci`	Logical; indicating whether or not error bars are to be plotted.
`bar`	Logical; specifies whether a barplot or just error bars should be shown.
`bar.col`	Color of bar.
`simlett`	A logical statement indicating whether or not letters should be shown above bars indicating that populations means have been determined to be significantly different.
`lett`	A vector of letters or some other code to display multiple comparison results.
`exp.fact`	A multiplication factor indicating how much extra room is made for drawing letters in top of graph. Only used if `simlett = TRUE`.
`xlab`	X axis label for plot.
`ylab`	Y axis label for plot.
`err`	The direction of error bars: `x` for horizontal, `y` for vertical.
`sfrac`	Scaling factor for the size of the "serifs" (end bars) on the confidence bars, in x-axis units.
`gap`	Size of gap in error bars around points (default 0; for `gap=TRUE` gives gap size of 0.01).
`slty`	Line type for error bars.
`scol`	Line color for error bars.
`pt.bg`	Background color of points. If `pch=NA`, no points are drawn (e.g. leaving room for text labels instead).
`order`	Logical, if `TRUE`, then treatments are ordered by their location statistics.
`names.arg`	A vector of names to be plotted below each bar or error bar. If this argument is omitted, then the names are taken from the names attribute of `y`.

`width`	Optional vector of bar widths.
`loc.meas`	Measure of location to be used for treatments, e.g. `mean` or `median`.
`cex.lab`	Size of axis labels
`cex.axis`	Expansion factor for numeric axis labels.
`cex.names`	Expansion factor for axis names (bar labels).
`las`	Style of axis labels. See `par`.
`horiz`	Logical value. If `FALSE`, then bars are drawn vertically with the first bar to the left. If `TRUE`, then bars are drawn horizontally with the first at the bottom.
`ylim`	Limits for Y-axis.
`...`	Additional arguments from `plotCI`.

Details

It is often desirable to display the results of a pairwise comparison procedure using sample measures of location and error bars. This functions allows these sorts of plots to be made. Note that this function is essentially a wrapper for the function `barplot` and `plotCI`. Much of the documentation here follows directly from B. Boklker and the documentation for `barplot`.

Value

A plot is returned.

Author(s)

Ken Aho created the wrapper for `barplot`. To create the function `plotCI` Ben Bolker documented and tweaked a function provided by Bill Venables.

References

Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983) *Graphical Methods for Data Analysis*. Wadsworth & Brooks/Cole. McGill, R., Tukey, J. W. and Larsen, W. A. (1978) Variations of box plots. *The American Statistician* 32, 12-16.

See Also

`barplot`, `plotCI`

Examples

```
eggs<-c(11,17,16,14,15,12,10,15,19,11,23,20,18,17,27,33,22,26,28)
trt<-c(1,1,1,1,1,2,2,2,2,2,3,3,3,3,4,4,4,4,4)
bplot(y=eggs, x=factor(trt),int="SE",xlab="Treatment",ylab="Mean number of eggs",
simlett=TRUE, lett=c("b","b","b","a"),pch=NA)
```

bromus — *Bromus tectorum dataset*

Description

Cheatgrass (*Bromus tectorum*) is an introduced annual graminoid that has invaded vast areas of sagebrush steppe in the intermountain west. Because it completes its vegetative growth stage relatively early in the summer, it leaves behind senescent biomass that burns easily. As a result areas with cheatgrass often experience a greater frequency of summer fires. A number of dominant shrub species in sagebrush steppe are poorly adapted to fire. As a result, frequent fires can change a community formerly dominated by shrubs to one dominated by cheatgrass. Nitrogen can also have a strong net positive effect on the cheatgrass biomass. A study was conducted at the Barton Road Long Term Experimental Research site (LTER) in Pocatello Idaho to simultaneously examine the effect of shrub removal and nitrogen addition on graminoid productivity.

Usage

```
data(bromus)
```

Format

The dataframe has 3 columns:

Plot Plot number.

Biomass Grass biomass in grams per meter squared.

Trt Treatment. C = Control, LN = Low nitrogen, HN = Hi Nitrogen, SR = Shrub removal.

bv.boxplot — *Bivariate boxplots*

Description

Creates diagnostic bivariate boxplots using the method of Goldberg and Iglewicz (1992). The boxplots can be used to check assumptions of bivariate normality. The default `robust=TRUE` option relies on on a biweight estimator function written by Everitt (2004).

Usage

```
bv.boxplot(X, Y, robust = TRUE, D = 7, xlab = "X", ylab = "Y")
```

Arguments

`X`	First of two quantitative variables making up the bivariate distribution.
`Y`	Second of two quantitative variables making up the bivariate distribution.
`robust`	Logical. Robust estimators, i.e. `robust = TRUE` are recommended.
`D`	The default `D = 7` lets the fence be equal to a 99 percent confidence interval for an individual observation.
`xlab`	Caption for *X* axis.
`ylab`	Caption for *Y* axis.

Details

Two ellipses are drawn. The inner is the "hinge" which contains 50 percent of the data. The outer is the "fence". Observations outside of the "fence" constitute possible troublesome outliers. The function `bivariate` from Everitt (2004) is used calculate the biweight *M*-estimators of location, scale and correlation if `robust = TRUE` (the default).

Value

A diagnostic plot is returned

Author(s)

Ken Aho, the function relies on an Everitt (2004) function for robust *M*-estimation.

References

Everitt, B. 2005. *An R and S-plus companion to multivariate analysis*. Springer.

Goldberg, K. M., and B. Ingelwicz. 1992. Bivariate extensions of the boxplot. *Technometrics* 34: 307-320.

See Also

`boxplot`

Examples

```
Y1<-rnorm(100,17,3)
Y2<-rnorm(100,13,2)
bv.boxplot(Y1,Y2)
```

`chi.plot` *Chi plots for diagnosing multivariate independence.*

Description

Chi-plots (Fisher and Switzer 1983, 2001) provide a method to diagnose multivariate non-independence among *Y* variables.

Usage

```
chi.plot(Y1, Y2)
```

Arguments

`Y1`	A *Y* variable of interest. Must be quantitative vector.
`Y2`	A second *Y* variable of interest. Must also be a quantitative vector.

Details

The method relies on calculating all possible pairwise differences within $\mathbf{y}_1$ and within $\mathbf{y}_2$. Let pairwise differences associated with the first observation in $\mathbf{y}_1$ that are greater than zero be transformed to ones and all other differences be zeros. Take the sum of the transformed values, and let this sum divided by (1 - *n*) be be the first element in the 1 x *n* vector **z**. Find the rest of the elements (2,..,*n*) in **z** using the same process.

Perform the same transformation for the pairwise differences associated with the first observation in $\mathbf{y}_2$. Let pairwise differences associated with the first observation in $\mathbf{y}_2$ that are greater than zero be transformed to ones and all other differences be zeros. Take the sum of the transformed values, and let this sum divided by (1 - *n*) be be the first element in the 1 x *n* vector **g**. Find the rest of the elements (2,..,*n*) in **g** using the same process.

Let pairwise differences associated with the first observation in $\mathbf{y}_1$ and the first observation in $\mathbf{y}_2$ that are both greater than zero be transformed to ones and all other differences be zeros. Take the sum of the transformed values, and let this sum divided by (1 - *n*) be be the first element in the 1 x *n* vector **h**. Find the rest of the elements (2,..,*n*) in **h** using the same process. We let:

$$S = sign((\boldsymbol{z} - 0.5)(\boldsymbol{g} - 0.5))$$

$$\chi = (\boldsymbol{h} - \boldsymbol{z} \times \boldsymbol{g})/\sqrt{\boldsymbol{z} \times (1 - \boldsymbol{z}) \times \boldsymbol{g} \times (1 - \boldsymbol{g})}$$

$$\lambda = 4 \times S \times max[(\boldsymbol{z} - 0.5)^2, (\boldsymbol{g} - 0.5)^2]$$

We plot the resulting paired χ and λ values for values of λ less than $4(1/(n-1) - 0.5)^2$. Values outside of $\frac{1.78}{\sqrt{n}}$ can be considered non-independent.

Value

Returns a chi-plot.

Author(s)

Ken Aho

References

Everitt, B. (2005) *R and S-plus companion to multivariate analysis*. Springer.

Fisher, N. I, and Switzer, P. (1985) Chi-plots for assessing dependence. *Biometrika*, 72: 253-265.

Fisher, N. I., and Switzer, P. (2001) Graphical assessment of dependence: is a picture worth 100 tests? *The American Statistician*, 55: 233-239.

See Also

`bv.boxplot`

Examples

```
Y1<-rnorm(100,15,2)
Y2<-rnorm(100,18,3.2)
chi.plot(Y1,Y2)
```

`ci.median` *Confidence interval for the median*

Description

Calculates the upper and lower confidence bounds for the true median, and calculates true coverage of the interval.

Usage

```
ci.median(x, conf = 0.95)
```

Arguments

`x`	A vector of quantitative data.
`conf`	The desired level of confidence 1 - *P*(type I error).

Value

Returns a list of `class = "ci"`. Default printed results are the paramter estimate and confidence bounds. Other `invisible` objects include:

`coverage`	The true coverage of the interval

.

Author(s)

Ken Aho

References

Ott, R. L., and M. T. Longnecker (2004) *A first course in statistical methods*. Thompson.

See Also

`median`

Examples

```
x<-rnorm(20)
ci.median(x)
```

`ci.mu.oneside` *One sided confidence interval for mu.*

Description

In some situations we may wish to quantify confidence in the region above or below a mean estimate. For instance, a biologist working with an endangered species may be interested in saying: "I am 95 percent confident that the true mean number of offspring is above a particular threshold". In a one-sided situation, we essentially let our confidence be 1- 2α (instead of 1 - α). Thus, if our significance level for a two-tailed test is α, our one-tailed significance level will be 2α.

Usage

```
ci.mu.oneside(data, conf = 0.95, n = NULL, Var = NULL, xbar = NULL,
summarized = FALSE, N = NULL, fpc = FALSE, tail = "upper")
```

Arguments

`data`	A vector of quantitative data. Required if `summarized=TRUE`.
`conf`	Level of confidence; 1 - P(type I error).
`n`	Sample size. Required if `summarized=TRUE`.
`Var`	Sample variance. Required if `summarized=TRUE`.
`xbar`	Sample mean. Required if `summarized=TRUE`.
`summarized`	Logical. Indicates whether summary statistics instead of raw data should be used.
`N`	Population size. Required if `summarized=TRUE`.
`fpc`	Logical. Indicating whether finite population corrections should be made.
`tail`	Indicates what side the one sided confidence limit should be calculated for. Choices are `"upper"` or `"lower"`.

Value

Returns a matrix with the sample mean and either the upper or lower confidence limit.

Author(s)

Ken Aho

References

Bain, L. J., and Engelhardt, M. (1992) *Introduction to probability and mathematical statistics*. Duxbury press. Belmont, CA, USA.

See Also

ci.mu.t

Examples

```
ci.mu.oneside(rnorm(100))
```

---

ci.mu.z *Z and t confidence intervals for mu.*

Description

These functions calculate t and z confidence intervals for μ. Z confidence intervals require specification (and thus knowledge) of σ. Both methods assume underlying normal distributions although this assumption becomes irrelevant for large sample sizes. Finite population corrections are provided if requested.

Usage

```
ci.mu.z(data, conf = 0.95, sigma = 1, summarized = FALSE, xbar = NULL,
fpc = FALSE, N = NULL, n = NULL)

ci.mu.t(data, conf = 0.95, summarized = FALSE, xbar = NULL, st.dev = NULL,
fpc = FALSE, N = NULL, n = NULL)
```

Arguments

data	A vector of quantitative data. Required if summarized = FALSE
conf	Confidence level; 1 - alpha.
sigma	The population standard deviation.

summarized	A logical statement specifying whether statistical summaries are to be used. If `summarized = FALSE`, then the sample mean and the sample standard deviation (`t.conf` only) are calculated from the vector provided in `data`. If `summarized = FALSE` then the sample mean `xbar`, the sample size `n`, and, in the case of `ci.mu.t`, the sample standard deviation `st.dev`, must be provided by the user.
xbar	The sample mean. Required if `summarized = TRUE`.
fpc	A logical statement specifying whether a finite population correction should be made. If `fpc = TRUE` then both the sample size `n` and the population size `N` must be specified.
N	The population size. Required if `fpc=TRUE`
st.dev	The sample standard deviation. Required if `summarized=TRUE`.
n	The sample size. Required if `summarized = TRUE`.

Details

`ci.mu.z` and `ci.mu.t` calculate confidence intervals for either summarized data or a dataset provided in `data`. Finite population corrections are made if a user specifies `fpc=TRUE` and provides some value for `N`.

Value

Returns a list of `class = "ci"`. Default printed results are the parameter estimate and confidence bounds. Other `invisible` objects include:

Margin	the confidence margin.

Author(s)

Ken Aho

References

Lohr, S. L. (1999) *Stampling: design and analysis*. Duxbury Press. Pacific Grove, USA.

See Also

`pnorm`, `pt`

Examples

```
#With summarized=FALSE
x<-c(5,10,5,20,30,15,20,25,0,5,10,5,7,10,20,40,30,40,10,5,0,0,3,20,30)
ci.mu.z(x,conf=.95,sigma=4,summarized=FALSE)
ci.mu.t(x,conf=.95,summarized=FALSE)
#With summarized = TRUE
ci.mu.z(x,conf=.95,sigma=4,xbar=14.6,n=25,summarized=TRUE)
ci.mu.t(x,conf=.95,st.dev=4,xbar=14.6,n=25,summarized=TRUE)
#with finite population correction and summarized = TRUE
ci.mu.z(x,conf=.95,sigma=4,xbar=14.6,n=25,summarized=TRUE,fpc=TRUE,N=100)
ci.mu.t(x,conf=.95,st.dev=4,xbar=14.6,n=25,summarized=TRUE,fpc=TRUE,N=100)
```

`ci.p` *Confidence interval estimation for the binomial parameter p.*

Description

Confidence interval formulae for μ are only appropriate for quantitative continuous variables. This of course excludes a large number of biologically important variables which describe binary outcomes or counts. The function `p.conf` calculates confidence intervals for the binomial parameter *p* (probability of success) using raw or summarized data. By default Wilson point estimators are used to estimate p and $\sigma_{\hat{p}}$. If raw data are to be used (the default) then successes should be indicated as ones and failures as zeros in the `data` vector. Finite population corrections can also be specified. Three methods for confidence intervals can be implemented: the normal approximation, Wilson estimators i.e. the adjusted Wald method (Wilson 1927), and the Clopper-Pearson exact method (Clopper and Pearson 1934). Agresti and Coull (1998) reccomend the Wilson estimation method.

Usage

```
ci.p(data, conf = 0.95, summarized = FALSE, phat = NULL, S.phat = NULL,
fpc = FALSE, n = NULL, N = NULL, method="wilson")
```

Arguments

`data`	A vector of binary data. Required if `summarized = FALSE`.
`conf`	Level of confidence 1 *P*(type I error).
`summarized`	Logical; indicate whether raw data or summary stats are to be used.
`phat`	Estimate of *p*. Required if `summarized = TRUE`.
`S.phat`	Estimate of $\sigma_{\hat{p}}$. Required if `summarized = TRUE`.
`fpc`	Logical. Indicates whether finite population corrections should be used. If `fpc = TRUE` then `N` must be specified. Finite population corrections are not possible for `method = "exact"`
`n`	Sample size. Required if `summarized = TRUE`.
`N`	Population size. Required if `fpc = TRUE`.
`method`	Type of method to be used in confidence interval calculations, `method ="wilson"` is the default, although there are two other options, `method="approximation"` provides the conventional normal approximation. `method="exact")` provides the Clopper Pearson (1934) method. The `exact` method cannot be implemented if `summarized=TRUE`.

Details

For the binomial distribution the parameter of interest is the probability of success, *p*. The parameter, *p*, and its standard deviation, σ_p , can be estimated with:

$$\hat{p} = \frac{x}{n},$$

$$S_{\hat{p}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

where *x* is the number of succeses and *n* is the number of observations.

Agresti and Coull (1998) reported that these estimators can create extremely inaccurate confidence intervals. As a result Ott and Longnecker (2004) recommend the Wilson estimators for estimation of *p* and $\sigma_{\hat{p}}$ (Wilson 1927).

$$\hat{p} = \frac{x+2}{n+4},$$

$$S_{\hat{p}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n+4}}$$

A 100(1 - α)% confidence interval for the binomial parameter *p* is found using:

$$\hat{p} \pm z_{1-(\alpha/2)}.$$

The "exact" method of Clopper and Pearson (1934) is bounded at the nominal limits, but actual coverage may greatly exceed nominal coverage. Confidence bounds for the Clopper and Pearson (1934) method are derived (in part) using quantiles from the *F*-distribution.

$$C_L = \frac{x}{(x + (n - x + 1)F^*_{1-\alpha/2}}$$

where $F^* \sim F(2n - 2x + 2, 2x)$

$$C_U = \frac{(x+1)F^*_{1-\alpha/2}}{n - x + (x+1)F^*_{1-\alpha/2}}$$

where $F^* \sim F(2x - 2, 2n - 2x)$.

Value

Returns a list of `class = "ci"`. Default printed results are the parameter estimate and confidence bounds. Other objects are `invisible`. In particular, if `method = "wilson" or "approximation"` returns a list with four items:

`p.hat`	Estimate for *p*.
`S.p.hat`	Estimate for $\sigma_{\hat{p}}$.
`margin`	Confidence margin.
`ci`	Confidence interval.

If `method = "wilson"` the function returns the confidence interval, `ci`, only, i.e. no other `invisible` components exist.

Note

This function contains only a few of the many methods that have been proposed for confidence interval estimation for *p*.

Author(s)

Ken Aho

References

Agresti, A., and Coull, B . A. (1998) Approximate is better than 'exact' for interval estimation of binomial proportions. *The American Statistician*. 52: 119-126.

Clopper, C. and Pearson, S. (1934) The use of confidence or fiducial limits illustrated in the case of the Binomial. *Biometrika* 26: 404-413.

Ott, R. L., and Longnecker, M. T. (2004) *A first course in statistical methods*. Thompson.

Wilson, E. B.(1927) Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association* 22: 209-212.

See Also

ci.mu.z, ci.p

Examples

```
#In 2001, it was estimated that 56,200 Americans would be diagnosed with
# non-Hodgkin's lymphoma and that 26,300 would die from it (Cernan et al. 2002).
# Here we find the 95% confidence interval for the probability of diagnosis, p.
ci.p(c(rep(0, 56200-26300),rep(1,26300)))
```

---

ci.sigma *Confidence interval for sigma squared.*

Description

The function calculates confidence intervals for σ^2. We assume that the parent population is normal.

Usage

```
ci.sigma(data, conf = 0.95, S.sq = NULL, n = NULL, summarized = FALSE)
```

Arguments

data	A vector of quantitative data. Required if summarized=FALSE.
conf	Level of confidence. 1 - *P*(type I error).
S.sq	Sample variance, required if summarized=TRUE.
n	Sample size, required if summarized=TRUE.
summarized	Logical. If summarized=TRUE then the user must supply S.sq and n

Value

Returns a list of `class = "ci"`. Default printed results are the parameter estimate and confidence bounds. Other objects are `invisible`.

Author(s)

Ken Aho

References

Bain, L. J., and M. Engelhardt. 1992. *Introduction to probability and mathematical statistics*. Duxbury press. Belmont, CA, USA.

See Also

`ci.mu.z`

Examples

```
ci.sigma(rnorm(20))
```

`ci.strat` *Confidence intervals for stratified random samples.*

Description

A statistical estimate along with its associated confidence interval can be considered to be an inferential statement about the sampled population. However this statement will only be a correct if the method of sampling is considered in the computations of standard errors. The function `ci.strat` provides appropriate computations given stratified random sampling.

Usage

```
ci.strat(data, strat, N.h, conf = 0.95, summarized = FALSE, use.t = FALSE,
n.h = NULL, x.bar.h = NULL, var.h = NULL)
```

Arguments

`data`	A vector of quantitative data. Required if `summarized=FALSE`.
`strat`	A vector describing strata.
`N.h`	A vector dexcribing the number of experimental units for each of the *k* strata.
`conf`	Level of confidence; 1 - *P*(type I error).
`summarized`	Logical. Indicates whether summarized data are to be used.
`use.t`	Logical. Indicates whether *t* or *z* confidence intervals should be built.

`n.h` A vector indicating the number of experimental units sampled in each of the *k* strata. Required if `summarized=TRUE`.

`x.bar.h` A vector containing the sample means for each of the *k* strata. Required if `summarized=TRUE` .

`var.h` A vector containing the sample variances for each of the *k* strata. Required if `summarized=TRUE`.

Details

the conventional formula for the sample standard error assumes simple random sampling. There are two other general types of sampling designs: stratified random sampling and cluster sampling. Since cluster sampling is generally used for surveys involving human demographics we will only describe corrections for stratified random sampling here. For more information on sample standard error adjustments for cluster sampling see Lohr (1999).

For a stratified random sampling design let *N* be the known total number of units in the defined population of interest, and assume that the population can be logically divided into *k* strata; $N = N_1 + N_2 + N_3 + \ldots + N_k$ (i.e. we are assuming that we know both the total population size, and the population size of each stratum). We sample each of the *k* strata with n_h observations; $h = 1, 2, \ldots, k$.

We estimate the variance in the *h*th stratum as:

$$S_h^2 = \frac{1}{n_h - 1}\sum_{i=1}^{n_k}(X_{hi} - \bar{X}_h)^2$$

where X_{hi} is the *i*th observation from the *h*th strata and $\bar{X}_h$ is the *h*th sample mean. We estimate the true population total, *T*, with:

$$\hat{T} = \sum_{h=i}^{k} N_h \bar{X}_h$$

We estimate the population mean, μ, with:

$$\bar{X}_{str} = \frac{\hat{T}}{N}$$

An unbiased estimator for the standard error of $\bar{X}_{str}$ is:

$$S_{\bar{X}_{str}} = \sqrt{\sum_{h=1}^{k}\left(1 - \frac{n_h}{N_h}\right)\left(\frac{N_h}{N}\right)^2\left(\frac{S_h^2}{n_h}\right)}$$

The standard error of $\hat{T}$ is also of interest. Here is an unbiased estimator.

$$S_{\hat{T}} = \sqrt{\sum_{h=1}^{k}\left(1 - \frac{n_h}{N_h}\right)N_h^2\left(\frac{S_h^2}{n_h}\right)}$$

Note that these standard errors have both a finite population correction and adjustments for stratification built into them. Assuming that sample sizes within each stratum are large or that the sampling design has a large number of strata, a 100(1 - α)percent confidence interval for μ and *T* can be constructed using:

$$\bar{X}_{str} \pm z_{1-\alpha/2} S_{\bar{X}_{str}}$$

$$\hat{T} \pm z_{1-\alpha/2} S_{\hat{T}}$$

In situations where sample sizes or the number of strata are small, a $t(n-k)$ distribution can (and should) be used for calculation of confidence intervals, where $n = n_1 + n_2 + \ldots + n_k$.

Value

Returns a list with two items:

`strat.summary`	A matrix with columns: `N.h`,`n.h`,`x.bar.h` and `var.h`
`CI`	Confidence intervals for μ and *T*

Author(s)

Ken Aho

References

Lohr, S. L. (1999) *Sampling: design and analysis*. Duxbury Press. Pacific Grove, USA.

Siniff, D. B., and Skoog, R. O. (1964) Aerial censusing of caribou using stratified random sampling. *Journal of Wildlife Management* 28: 391-401.

See Also

`ci.mu.z`

Examples

```
#Data from Siniff and Skoog (1964)
Caribou<-data.frame(Stratum=c("A","B","C","D","E","F"),N.h=c(400,30,61,18,70,120),
n.h=c(98,10,37,6,39,21),x.bar.h=c(24.1,25.6,267.6,179,293.7,33.2),
var.h=c(5575,4064,347556,22798,123578,9795))
attach(Caribou)
ci.strat(data,strat=Stratum,N.h=N.h,conf=.95,summarized=TRUE,use.t=FALSE,n.h=n.h,
x.bar.h=x.bar.h,var.h=var.h)
```

cliff.env	*Environmental data for the community dataset cliff.sp*

Description

The data here are a subset of a dataset collected by Aho (2006) which describe the distribution of communities of lichens and vascular and avascular plant species on montane cliffs in Northeast Yellowstone National Park. Of particular interest was whether substrate (limestone or andesitic conglomerate) or water supply influenced community composition.

Usage

```
data(cliff.env)
```

Format

This data frame contains the following columns:

sub a factor with 2 levels `"Andesite"` and `"Lime"` describing substrate type.

water a factor with 3 levels `"W"` `"I"` `"D"` indicating wet, intermediate, or dry conditions.

Details

Two categorical environmental variables are described for 54 sites. `sub` describes the substrate; there are two levels: `"Andesite"` and `"Lime"`. `water` describes distance of samples from waterfalls which drain the cliff faces; there are three levels `"W"` indicating wet, `"I"` indicating intermediate, and `"D"` indicating dry.

Source

Aho, K.(2006) *Alpine ecology and subalpine cliff ecology in the Northern Rocky Mountains*. Doctoral dissertation, Montana State University, 458 pgs.

cliff.sp	*Yellowstone NP cliff community data*

Description

A subset of a dataset collected by Aho (2006) which describes the distribution of communities of lichens and vascular and avascular plant species on montane cliffs in Northeast Yellowstone National Park. Of particular interest was whether substrate (limestone or andesitic conglomerate) or water supply influenced community composition.

Usage

```
data(cliff.sp)
```

Details

Responses are average counts from two 10 x 10 point frames at 54 sites. Abundance data are for eleven species, 9 lichens, 3 mosses, and 2 vascular plants. Data were gathered in the summer of 2004 on two andesitic/volcanic peaks (Barronette and Abiathar) with sedimentary layers at lower elevations.

Source

Aho, K.(2006) *Alpine ecology and subalpine cliff ecology in the Northern Rocky Mountains*. Doctoral dissertation, Montana State University, 458 pgs.

ConDis.matrix — *Calculation and display of concordant and discordant pairs*

Description

Calculates whether pairs of observations from two vectors are concordant discordant or neither. These are displayed in the lower diagonal of a symmetric output matrix as 1, -1 or 0.

Usage

```
ConDis.matrix(Y1, Y2)
```

Arguments

`Y1`	A vector of quantitative data.
`Y2`	A vector of quantitative data. Observations are assumed to be paired with respective observations from `Y1`.

Details

Consider all possible combinations of (Y_{1i}, Y_{ij}) and (Y_{2i}, Y_{ij}) where $1 <= i < j <= n$. A pair is concordant if $Y_{1i} > Y_{1j}$ and $Y_{2i} > Y_{2j}$ or if $Y_{1i} < Y_{1j}$ and $Y_{2i} < Y_{2j}$. Conversely, a pair is discordant if $Y_{1i} < Y_{1j}$ and $Y_{2i} > Y_{2j}$ or if $Y_{1i} > Y_{1j}$ and $Y_2i < Y_2j$. This information has a number of important uses including calculation of Kendall's τ.

Value

A matrix is returned. The lower triangle indicates whether observations are concordant (element = 1), disconcordant (element = -1) or neither (element = 0).

Author(s)

Ken Aho

References

Hollander, M., and Wolfe, D. A. (1999) *Nonparametric statistical methods*. New York: John Wiley & Sons.

Liebetrau, A. M. (1983) *Measures of association*. Sage Publications, Newbury Park, CA.

Sokal, R. R., and Rohlf, F. J. (1995) *Biometry*. W. H. Freeman and Co., New York.

See Also

`cor`

Examples

```
#Crab data from Sokal and Rohlf (1998)
crab<-data.frame(gill.wt=c(159,179,100,45,384,230,100,320,80,220,320,210),
body.wt=c(14.4,15.2,11.3,2.5,22.7,14.9,1.41,15.81,4.19,15.39,17.25,9.52))
attach(crab)
crabm<-ConDis.matrix(gill.wt,body.wt)
crabm
```

---

const — *Constancy of species in a community dataset*

Description

Calculates constancy of species in a community data (site x species matrix) with respect to categorical treatments (e.g. cluster analysis classes). Constancy of a species to a treatment is the proportion of times the species occurs at sites within the treatment.

Usage

```
const(Y, cat, digits = 4)
```

Arguments

`Y`	An n x p community site x species matrix.
`cat`	An n x 1 vector of categorical assignments.
`digits`	Number of significant digits in output.

Value

Output is a p x r matrix where r is the number of categorical levels in `cat`.

Author(s)

Ken Aho

See Also

fidelity, veg.table

Examples

```
library(vegan)
data(dune)
data(dune.env)
const(dune,dune.env[,3])
```

D.sq *Mahalanobis distance for two sites using a pooled covariance matrix*

Description

Allows much easier multivariate comparisons of groups of sites then provided by the function mahalanobis in the base library.

Usage

```
D.sq(g1, g2)
```

Arguments

g1	Community vector for site 1
g2	Community vector for site 2

Author(s)

Ken Aho

References

Legendre, P, and L. Legendre (1998) *Numerical ecology, 2nd English edition*. Elsevier, Amsterdam, The Netherlands.

See Also

mahalanobis

Examples

```
g1<-matrix(ncol=3,nrow=3,data=c(1,0,3,2,1,3,4,0,2))
g2<-matrix(ncol=3,nrow=3,data=c(1,2,4,5,2,3,4,3,1))
D.sq(g1,g2)$D.sq
```

`deer` *Maternal deer data*

Description

Monteith et al. (2009) examined the maternal life history characteristics of white-tailed deer (Odocoileus virginianus) originating from the Black Hills in southwestern South Dakota and from eastern South Dakota. Because litter size and dam size affects offspring weight the investigators used proportional birth mass (dam mass/total litter mass) as a measure of reproductive investment by deer.

Usage

```
data(deer)
```

Format

The dataframe contains 6 columns

Birth.Yr Year of birth.

Litter.size Number of offspring.

Region Categorical variable with two factor levels. BH = Black Hills, ER = Eastern Region.

Dam.weight Dam weight in kg.

Total.birth.mass Mass of litter in kg.

Prop.mass Total birth mass divided by dam weight.

Source

Monteith, K. L, Schmitz, L. E., Jenks, J. A., Delger, J. A., and R. T. Bowyer. 2009. Growth of male white tailed deer: consequences of maternal effects. *Journal of Mammalogy* 90(3): 651-660.

`deer.296` *Mule deer telemetry data*

Description

Telemetry data for mule deer #296 from the Starkey Experimental Forest in Northeastern Oregon. Data are high resolution (10 minute) radio collar readings from 8/20/2008 to 11/6/2008. Also included are data for nearest neighbor locations of forest/grassland boundaries.

Usage

```
data(deer.296)
```

Format

A data frame with 5423 observations on the following 7 variables.

`Time` Unit of time measurement used at the Starkey Experimental Forest

`X` Mule Deer X-coordinate, UTM Easting

`Y` Mule Deer Y-coordinate, UTM Northing

`NEAR_X` Nearest boundary location X coordinate

`NEAR_Y` Nearest boundary location Y coordinate

`Hab_Type` Type of habitat

`NEAR_ANGLE` A numeric vector containing the angle of azimuth to the nearest point on the boundary with respect to a four quadrant system. NE = $0°$ to $90°$, NW is > $90°$ and $\leq 180°$, SE is < $0°$ and $\leq -90°$ is > $-90°$ and $\leq -180°$.

`demos` *Pulldown menu for 'asbio' interactive graphical functions.*

Description

Provides a pulldown GUI menu to allow access to **asbio** statistical and biological graphical demos, e.g. `anm.ci`, `samp.dist`, `loglik.plot`, etc. The function currently provides links to over 100 distinct GUIs (many of these are slaves to other GUIs), which in turn provide distinct graphical demonstrations. The GUIs work best with an SDI R interface.

Usage

```
demos()
```

Author(s)

Ken Aho

See Also

`anm.coin`, `anm.ci`, `anm.die`, `anm.exp.growth`, `anm.geo.growth`, `anm.log.growth`, `loglik.plot`, `anm.LVcomp`, `anm.LVexp`, `anm.ls`, `anm.transM`, `lmu.tck`, `samp.dist`, `see.HW`, `see.logic`, `see.M`, `see.nlm`, `see.norm.tck`, `see.power`, `see.regression`, `see.typeI_II`, `selftest.se.tck1`, `shade.norm`, `Venn`

Examples

```
## Not run:
demos()

## End(Not run)
```

DH.test *Doornick-Hansen test for multivariate normality.*

Description

The Doornick-Hansen test for multivariate normality is a powerful alternative to the Shapiro-Wilk test.

Usage

```
DH.test(Y, Y.names = NULL)
```

Arguments

`Y`	An n x p matrix of dependent variables.
`Y.names`	Names of `Y` variables; should be a 1 x p character string.

Details

An assumption of multivariate normality is exceedingly difficult to verify. Hypothesis tests tend to be too stringent, and multivariate diagnostic plots only allow viewing of two variables at a time. Univariate normality of course can be verified using normal probability plots. However while marginal non-normality indicates multivariate non-normality, marginal normality does not insure that Y variables collectively follow a multivariate normal distribution.

The Doornik-Hansen test for multivariate normality (Doornik and Hansen 2008) is based on the skewness and kurtosis of multivariate data that is transformed to insure independence. The DH test is more powerful than the Shapiro-Wilk test for most tested multivariate distributions (Doornik and Hansen 2008). The function `DH.test` also runs the Doornik-Hansen test for both multivariate and univariate normality. The later test follows directly from the work of Bowman and Shenton (1975), Shenton and Bowman (1977) and D'Agostino (1970).

Value

Returns a list with two objects.

`multi`	A dataframe containing multivariate results, i.e. the test statistic, E, the degrees of freedom and the p-value.
`comp2`	A dataframe with p rows detailing univariate tests. Columns in the dataframe contain the test statistics, degrees of freedom and p-values.

Note

As with all inferential normality tests our null hypothesis is that the underlying population is normal, in this case multivariate normal.

Author(s)

Ken Aho

References

D'Agostino, R. B. (1970). Transformation to normality of the null distribution of g1, Biometrika, 57: 679-681.

Doornik, J.A. and Hansen, H. (2008). An Omnibus test for univariate and multivariate normality. Oxford Bulletin of Economics and Statistics, 70, 927-939.

Shenton, L. R. and Bowman, K. O. (1977). A bivariate model for the distribution of b1 and b2, Journal of the American Statistical Association, 72: 206.211.

See Also

`shapiro.test`, `bv.boxplot`

Examples

```
data(iris)#The ubiquitous multivariate iris data.
DH.test(iris[,1:4],Y.names=names(iris[,1:4]))
```

`e.cancer` *Esophageal cancer data modified slightly to create a balanced three-way factorial design*

Description

Breslow and Day (1980) studied the effect of age, tobacco, and alcohol on esophageal cancer rates at Ile-et-Vilaine, France. Data are altered slightly to make the design balanced, and to allow enough degrees of freedom to perform a fully factorial three way ANOVA.

Usage

```
data(e.cancer)
```

Format

The dataset contains four variables:

age grp. age group, a factor with four levels: `"25-34"`, `"35-44"`, `"45-54"`, `"55-64"`, and `"65-74"`.

alcohol alcohol consumed (g/day).

tobacco tobacco consumed (g/day).

cases number of esophageal cancer cases.

Source

Breslow, N. E. and N. E. Day. 1980. Statistical Methods in Cancer Research. 1: The Analysis of Case-Control Studies. IARC Lyon / Oxford University Press.

`ES.May` *May's effective specialization index*

Description

May and Beverton (1990) created the effective specialization index to quantify the degree of specialization of insects with potential host plants.

Usage

```
ES.May(mat, digs = 3)
```

Arguments

`mat`	A symmetric matrix with potential specialist hosts in rows and and the number species specializing on each of the host species in columns (see details below).
`digs`	The number of significant digits in output.

Details

The structure of the object `mat` is nonintuitive. In the rows of the matrix are species which can be selected by potential specialists (i.e. hosts). May and Beverton (1990) used four oak species. The columns indicate the degree of specialization of potential specialists. May and Beverton (1990) were interested in the specialization of beetles. The first element (row 1, column 1) in their 4 x 4 matrix contained only beetle species found on host 1. The second element (row 1, column 2) contained the number of beetle species found on host 1 and one other host. The third element (row 1, column 3) contained the number of beetle species found on host 1 and two other hosts. The fourth element (row 1, column 4) contained the number of beetle species occurring on all four hosts.

Value

Output is a list

`E.S_coefficients`

`Nk`	The number of distinct specialists.
`Pki.matrix`	The proportion of potential specialists on the *k*th host
`N.matrix`	The raw data.
`fk.matrix`	
`fk.vector`	For the *k*th host, the proportion of species which are effectively specialized.
`Nk.vector`	The number of species which are effectively specialized on the *k*th host.

Author(s)

Ken Aho and Jessica Fultz

References

May, R. M. and Beverton, R. J. H. (1990) How many species [and discussion]. *Philosophical Transactions: Biological Sciences*. 330 (1257) 293-304.

Examples

```
#data from May and Beverton (1990)
beetle<-matrix(ncol=4,nrow=4,data=c(5,8,7,8,20,10,9,8,14,15,11,8,15,15,12,8),
byrow=TRUE)
ES.May(beetle)
```

`evaluators` *Cluster analysis evaluators.*

Description

A collection of classification evaluators. `McR.eval` provides both the McClain & Rao evaluator (W/B) (McClain and Rao 1975) and the PARTANA ratio (Roberts 2005); `Cindex.eval` = The *C*-index (Hubert and Levin 1976); `morisita.eval` = the Morisita index (adapted from Horn 1966); `biserial.eval` = point biserial correlation evaluator (Brogden 1949).

Usage

```
McR.eval(cat, dist, method = "McR")

Cindex.eval(cat, Y, index = "steinhaus")

morisita.eval(cat, Y)

biserial.eval(cat, dist)
```

Arguments

`cat`	Classification solution, a categorical vector.
`dist`	A dissimilarity or distance matrix, i.e. an object of `class="dist"`.
`method`	The method used in `McR.eval`. Options are `method="partana"` and `method="McR"`.
`Y`	A matrix of raw data, e.g. a community matrix.
`index`	Type of dissimilarity or distance metric to use. Any measure from `get.dist` is allowed

Details

More to come. Note that the function `McR.eval` is essentially the `partana` function from library `labdsv` with only a few minor adjustments.

Value

Returns an list of `class="eval"`. Printed will be the evaluator score for a classification solution; `invisible` objects will vary with method.

Note

The Morisita evaluator has been tested on a few datasets and appears to respond in a strongly linear fashion to the number of clusters.

Author(s)

David Roberts and Ken Aho

References

Brogden, H. E.(1949) A new coefficient: application to biserial correlation and to estimation of selective efficiency. *Psychometrika*. 14: 169-182.

Horn, H. S. (1966) Measurement of "overlap" in comparative ecological studies. *Am. Naturalist*. 100: 419-424.

Hubert, L. J. & Levin, J. R. (1976) A general framework for assessing categorical clustering in free recall. *Psychol. Bull*. 83: 1072-1080.

McClain, J. O. & Rao, V. R. (1975) CLUSTISZ: A program to test for the quality of clustering of a set of objects. *J. Marketing Res*. 12: 456-460.

Milligan, G. W. (1981) A Monte Carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*. 46(2): 187-199.

Milligan, G. W. & Cooper, M. C. (1985) An examination of procedures for determining the number of clusters in a dataset. *Psychometrika*. 50 (2): 159-179.

Milligan, G. W. & Isaac, P. D. (1980) The validation of four ultrametric clustering algorithms. *Pattern Recogn*. 12: 41-50.

Roberts, D. (2005) *Vegetation classification in R, for labdsv ver. 1.1-1, vegetation ecology package*. `www.cran.r-project.org`. unpubl.

See Also

`get.dist`

Examples

```
library(vegan)
data(dune)
data(dune.env)
```

```
McR.eval(dune.env[,3],get.dist(dune,"steinhaus"))
Cindex.eval(dune.env[,3],dune)
biserial.eval(dune.env[,3],get.dist(dune,"steinhaus"))
```

---

evenness *Pielou's measure of species evenness*

Description

Calculates Pielou's measure of species evenness, i.e. $J = H'/\ln(S)$ where H' is Shannon Weiner diversity and S is the total number of species in a sample, across all samples in dataset.

Usage

```
evenness(x)
```

Arguments

x	A vector or matrix of species abundances (e.g. counts). The function assumes that species are in columns and sites are in rows.

Details

Many diversity indices incorporate evenness (e.g. Simpson's diversity, Shannon- Weiner diversity). Diversity indices which concentrate totally on evenness are fraught with problems including dependence on species counts (McCune and Grace 2002). A particular problem with Pielou's index is that it is a ratio of a relatively stable index, H', and one that is strongly dependent on sample size, S.

Value

Returns Pielou's J.

Author(s)

Ken Aho

References

McCune, B., and Grace, J.B. (2002) *Analysis of ecological communities*. MjM Software design. Gelenden Beach OR.

See Also

SW.index, fidelity, const

Examples

```
library(vegan)
data(varespec)
evenness(varespec)
```

`exercise.repeated` *Repeated measures data for an exercise experiment.*

Description

Freund et al. (1986) listed data for a longitudinal study of exercise therapies. The data were analyzed using AR1 covariance matrices in mixed models by Fitzmaurice et al. (2004). In the study 37 patients were randomly assigned to one of two weightlifting programs. In the first program (TRT 1), repetitions with weights were increased as subjects became stronger. In the second program (TRT 2), the number of repetitions was fixed but weights were increased as subjects became stronger. An index measuring strength was created and recorded at day 0, 2, 4, 6, 8, 10, and 12.

Usage

```
data(exercise.repeated)
```

Format

The dataframe contains a repeated measures dataset describing the strength of 37 subjects with respect to two weightlifting programs. There are four columns:

ID Subject ID.

TRT The type of weightlifting treatment (a factor with two levels, `1` and `2`).

strength A strength index.

day The day that `strength` was measured on a subject, measured from the start of the experiment.

Source

Fitzmaurice, G. M., Laird, N. M, and Waird, J. H. (2004) *Applied longitudinal analysis*. Wiley.

fidelity — *Fidelity of species in a community to a particular group*

Description

The function `fidelity` calculates the proportion of experimental units (sites) in a group a species occurs in, compared to the total number of sites the species occurs in across all groups.

Usage

```
fidelity(Y, cat, digits = 4)
```

Arguments

`Y`	An n x p community matrix.
`cat`	An n x 1 vector of categorical assignments.
`digits`	The number of significant digits in output

Value

Returns a p x r matrix of species fidelities (where r is the number of categorical assignments, e.g. factor levels.)

Author(s)

Ken Aho

See Also

`const`, `evenness`, `veg.table`

Examples

```
library(vegan)
data(dune)
data(dune.env)
fidelity(dune,dune.env[,3])
```

`FR.multi.comp` *Multiple pairwise comparison procedure to accompany a Friedman test.*

Description

As with ANOVA we can examine multiple pairwise comparisons from a Freidman test after we have rejected our omnibus null hypothesis. However we will need to account for the fact that these comparisons will be non-orthogonal. A conservative multiple comparison method used here is based on the Bonferroni procedure.

Usage

```
FR.multi.comp(Y, X, blocks, nblocks, conf = 0.95)
```

Arguments

`Y`	A vector of responses, i.e. quantitative data.
`X`	A categorical vector of factor levels.
`blocks`	A categorical vector of blocks.
`nblocks`	The number of blocks.
`conf`	The level of confidence. 1 - *P*(type I error).

Value

Returns a six column dataframe containing:

1) the type of contrast (names are taken from levels in `X`),

2) the mean rank difference,

3) the lower confidence bound of the true mean rank difference,

4) the upper confidence bound of the true mean rank difference,

5) the hypothesis decision rule given the prescribed significance level, and

6) the adjusted *p*-value.

Author(s)

Ken Aho

References

Fox, J. R., and Randall, J. E. (1970) Relationship between forearm tremor and the biceps electromyogram. *Journal of Applied Physiology* 29: 103-108.

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

friedman.test

Examples

```
#Data from Fox and Randall (1970)
Tremors<-data.frame(freq=c(2.58,2.63,2.62,2.85,3.01,2.7,2.83,3.15,3.43,3.47,2.78,
2.71,3.02,3.14,3.35,2.36,2.49,2.58,2.86,3.1,2.67,2.96,3.08,3.32,3.41,2.43,2.5,
2.85,3.06,3.07),
weights=factor(rep (c(7.5,5,2.5,1.25,0), 6)),block=factor(rep (1:6,each=5)))
attach(Tremors)
FR.multi.comp(Y=freq,X=weights,blocks=block,nblocks=6, conf=.95)
```

G.mean *Geometric mean*

Description

Calculates the geometric mean.

Usage

```
G.mean(x)
```

Arguments

x	A vector of quantitative data.

Value

Returns the geometric mean.

Author(s)

Ken Aho

See Also

H.mean, HL.mean

Examples

```
x<-c(2,1,4,5,6,2.4,7,2.2,.002,15,17,.001)
G.mean(x)
```

G.stat *G-tests*

Description

Performs *G*-tests for contingency table analyses.

Usage

```
G.stat(y)
```

Arguments

y	A matrix containing a contingency table.

Details

In two-way tables the data are arranged so that the *c* factor levels from the explanatory variable are in columns and the *r* response variable categories are in rows. Experimental unit responses are counts within the contingency table.

Value

G	The *G* test statistic (i.e. 2*the likelihood ratio).
lik.ratio.p.value	P-value for the test.

Author(s)

Ken Aho

References

Zar, J. H. (1999) *Biostatistical Analysis, 4th ed.* Prentice-Hall.

Examples

```
jugularis<-matrix(nrow=2,ncol=2,data=c(20,0,29,21),byrow=TRUE)
G.stat(jugularis)
```

`get.dist` *Calculates 26 possible resemblance measures.*

Description

This function allows users access to 26 resemblance measures . This includes five not previously available in R. Many of the measures are programmed with the function `designdist` from library **vegan**. The methods can be specified by name or by the distance measure number they were given in Table 29.10 in (Aho; in prep); e.g. Steinhaus index = `"D8"`. Minkowski's distance requires an additional specification for power. The default is power = 2 which makes the measure equivalent to Euclidean distance.

Usage

```
get.dist(data, method, minkowski.power = 2)
```

Arguments

`data` A matrix for which resemblances between rows will be calculated

`method` One of twenty six possible resemblance measures. These are: `"matching"`, `"rogers"`, `"jaccard.pa"`, `"sorenson"`, `"kulkczynski.pa"`, `"ochiai"`, `"gower"`, `"steinhaus"`, `"kulkczynski.q"`, `"jaccard.q"`, `"euclidean"`, `"rel.euclidean"`, `"manhattan"`, `"czekanowski"`, `"whittaker"`, `"canberra"`, `"chi.metric"`,`"chi.dist"`, `"morisita"`, `"morisita.horn"`, `"minkowski"`, `"mountford"`, `"raup.crick"`, `"binomial"`, or `"chao"`. A method may also be specified by its equation number from Ch. 29. i.e. Steinhaus dissimilarity = `"D8"`.

`minkowski.power` Minkowski's distance requires a specification for power. The default is `minkowski.power = 2` which makes the measure equivalent to Euclidean distance.

Details

See Ch. 29 for descriptions of resemblance measures.

Value

Returns a matrix of `class(dist)`.

Author(s)

Ken Aho

References

Aho, K. In prep. *Applied statistics for biologists, a textbook using R*.

Legendre, P, and Legendre, L. (1998) *Numerical ecology, 2nd English edition*. Elsevier, Amsterdam, The Netherlands.

Oksanen, J., Kindt, R., Legendre, P., O'Hara, B., Simpson, G. L., and Stevens, M. H. H. (2008) *vegan: community ecology package. R package version 1.13-0*. http://vegan.r-forge.r-project.org

See Also

dist

Examples

```
library(vegan)
data(varespec)
get.dist(varespec,method="steinhaus")
```

Glucose2 *Glucose Levels Following Alcohol Ingestion*

Description

The Glucose2 data frame has 196 rows and 4 columns.

Format

This data frame contains the following columns:

Subject a factor with levels 1 to 7 identifying the subject whose glucose level is measured.

Date a factor with levels 1 2 indicating the occasion in which the experiment was conducted.

Time a numeric vector giving the time since alcohol ingestion (in min/10).

glucose a numeric vector giving the blood glucose level (in mg/dl).

Details

Hand and Crowder (Table A.14, pp. 180-181, 1996) describe data on the blood glucose levels measured at 14 time points over 5 hours for 7 volunteers who took alcohol at time 0. The same experiment was repeated on a second date with the same subjects but with a dietary additive used for all subjects.

Note

descriptions and details are from the library nlme.

Source

Pinheiro, J. C. and Bates, D. M. (2000), *Mixed-Effects Models in S and S-PLUS*, Springer, New York. (Appendix A.10)

Hand, D. and Crowder, M. (1996), *Practical Longitudinal Data Analysis*, Chapman and Hall, London.

H.mean *Harmonic mean*

Description

Calculates the harmonic mean.

Usage

```
H.mean(x)
```

Arguments

x	Vector of quantitative data.

Value

Returns the harmonic mean.

Author(s)

Ken Aho

See Also

G.mean, HL.mean

Examples

```
x<-c(2,1,4,5,6,2.4,7,2.2,.002,15,17,.001)
H.mean(x)
```

`HL.mean` *Hodges-Lehman estimator of location*

Description

Calculates Hodges-Lehman estimates of location.

Usage

```
HL.mean(x)
```

Arguments

x	A vector of quantitative data.

Author(s)

Ken Aho

See Also

H.mean, G.mean

Examples

```
x<-c(2,1,4,5,6,2.4,7,2.2,.002,15,17,.001)
HL.mean(x)
```

`Hotelling` *Hotelling T-squared test*

Description

The Hotelling T-squared test provides a multivariate analog for a univariate test of two populations.

Usage

```
Hotelling(Y, X)
```

Arguments

Y	An n x p matrix of quantitative Y variables.
X	A n x 1 vector describing a factor with two factor levels.

Details

Details are provided in any introductory text on multivariate statistics. The same result will be given if a one way MANOVA is run on the data.

Value

Returns a list

`D.sq`	The Mahalanobis distance between the factor levels based on a pooled covariance matrix.
`T.sq`	The T^2 test statistic
`table`	A table with the F-statistic summary including the test statistic, numerator and denominator degrees of freedom, and the p-value

Author(s)

Ken Aho

References

Everitt, B. (2005) *An R and S-plus companion to multivariate analysis*. Springer.

See Also

`manova`

Examples

```
Y1<-rnorm(100,15,3)
Y2<-rnorm(100,17,2)
X<-factor(c(rep(1,50),rep(2,50)))

Hotelling(cbind(Y1,Y2),X)
anova(lm(cbind(Y1,Y2)~X))
```

`huber.mu` *Huber M-estimator of location*

Description

The Huber M-estimator is a robust high efficiency estimator of location that has probably been under-utilized by biologists. It is based on maximizing the likelihood of a weighting function. This is accomplished using an iterative least squares process. The Newton Raphson algorithm is used here. The function usually converges fairly quickly (< 10 iterations). The function uses the Median Absolute Deviation function, `mad`, from `MASS`. Note that if MAD = 0, then `NA` is returned.

Usage

```
huber.mu(x, c = 1.28, iter = 20, conv = 1e-07)
```

Arguments

`x`	A vector of quantitative data.
`c`	Stop criterion. The value `c = 1.28` gives 95 percent efficiency of the mean given normality.
`iter`	Maximum number of iterations.
`conv`	Convergence criterion.

Value

Returns Huber's *M*-estimator of location.

Author(s)

Ken Aho

References

Huber, P. J. (2004) *Robust Statistics*. Wiley.

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`huber.one.step`, `huber.NR`

Examples

```
x<-rnorm(100)
huber.mu(x)
```

`huber.NR` *Huber M-estimator iterative least squares algorithm*

Description

Algorithm for calculating fully iterated or one step Huber *M*-estimators of location.

Usage

```
huber.NR(x, c = 1.28, iter = 20)
```

Arguments

`x`	A vector of quantitative data
`c`	Bend criterion. The value `c = 1.28` gives 95 percent efficiency of the mean given normality.
`iter`	Maximum number of iterations

Details

The Huber *M*-estimator is a robust high efficiency estimator of location that has probably been under-utilized by biologists. It is based on maximizing the likelihood of a weighting function. This is accomplished using an iterative least squares process. The Newton Raphson algorithm is used here. The function usually converges fairly quickly < 10 iterations. The function uses the Median Absolute Deviation function, `mad`, from `MASS`. Note that if MAD = 0, then `NA` is returned.

Value

Returns iterative least squares iterations which converge to Huber's *M*-estimator. The first element in the vector is the sample median. The second element is the Huber one-step estimate.

Author(s)

Ken Aho

References

Huber, P. J. (2004) *Robust Statistics*. Wiley.

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`huber.one.step`, `huber.mu`

Examples

```
x<-rnorm(100)
huber.NR(x)
```

`huber.one.step` *Huber one step M-estimator*

Description

Returns the first Raphson-Newton iteration of the function `Huber.NR`.

Usage

```
huber.one.step(x, c = 1.28)
```

Arguments

`x`	Vector of quantitative data
`c`	Bend criterion. The value `c = 1.28` gives 95 percent efficiency of the mean given normality.

Details

The Huber *M*-estimator function usually converges fairly quickly, hence the justification of the Huber one step estimator. The function uses the Median Absolute Deviation function, `mad`, from `MASS`. If MAD = 0, then `NA` is returned.

Value

Returns the Huber one step estimator.

Author(s)

Ken Aho

References

Huber, P. J. (2004) *Robust Statistics*. Wiley.

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`huber.mu`, `huber.NR`

Examples

```
x<-rnorm(100)
huber.one.step(x)
```

`illusions` *Visual illusions illustrating human preception errors.*

Description

In development, currently displays three illusions. Illusion 3 is from Yihui Xie's package **animation**.

Usage

```
illusions(ill.no = 1)
```

Arguments

`ill.no`	Numeric describing which illusion number to view.

Author(s)

Ken Aho

`island.rich` *Mammal richness data with respect to montane island area.*

Description

Among other things the theory of island biogeography (Wilson and McArthur 1967) predicts that species richness will increase with increasing island area. Islands, however, need not be a body of land in water. Lomolino et al. (1989) investigated the relationship between the area of montane forest patches and the richness of mammal fauna in the Southwestern United States. His data are compiled here.

Usage

```
data(island.rich)
```

Format

This data frame contains the following columns:

Island a factor with the montane island names.

Richness a numeric vector providing the the number of mammal species identified per island.

Area_km2 Montane island area in (in km^2).

Source

Lomolino, M. V., J. H. Brown and R. Davis (1989) Island biogeography of montane forest mammals in the American Southwest. *Ecology* 70:180-194.

joint.confint   *Calculates joint confidence intervals for parameters in linear models using a Bonferroni procedure.*

Description

Creates widened confidence intervals to allow joint consideration of parameter confidence intervals.

Usage

```
joint.confint(model, conf = 0.95)
```

Arguments

`model`	A linear model created by `lm`
`conf`	level of confidence 1 - *P*(type I error)

Details

As with all Bonferroni-based methods for joint confidence the resulting intervals are exceedingly conservative and thus are prone to type II error.

Value

Returns a dataframe with the upper and lower confidence bounds for each parameter in a linear model.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

`confint`, `p.adjust`

Examples

```
Soil.C<-c(13,20,10,11,2,25,30,25,23)
Soil.N<-c(1.2,2,1.5,1,0.3,2,3,2.7,2.5)
Slope<-c(15,14,16,12,10,18,25,24,20)
Aspect<-c(45,120,100,56,5,20,5,15,15)
Y<-as.vector(c(20,30,10,15,5,45,60,55,45))
model<-lm(Y~Soil.C+Soil.N+Slope+Aspect)
joint.confint(model)
```

Kappa — *Calculates kappa statistic and other classification error statistics*

Description

The kappa statistic, along with user and producer error rates are conventionally used in the remote sensing to describe the effectiveness of ground cover classifications. Since it simultaneously considers both errors of commission and omission, kappa can be considered a more conservative measure of classification accuracy than the percentage of correctly classified items.

Usage

```
Kappa(class1, reference)
```

Arguments

`class1`	A vector describing a classification of experimental units.
`reference`	A vector describing the "correct" classification of the experimental units in `class1`

Value

Returns a list with 5 items

`ttl_agreement`	The percentage of correctly classified items.
`user_accuracy`	The user accuracy for each category of the classification.
`producer_accuracy`	The producer accuracy for each category of the classification.
`table`	A two way contingency table comparing the user supplied classification versus the reference classification.

Author(s)

Ken Aho

References

Jensen, J. R. (1996) *Introductory digital imagery processing 2nd edition*. Prentice-Hall.

Examples

```
reference<-c("hi","low","low","hi","low","med","med")
class1<-c("hi","hi","low","hi","med","med","med")
Kappa(class1,reference)
```

---

km *Kaplan-Meier survivorship.*

Description

Calculates survivorship for individuals in a population over time based on the method of Kaplan-Meier; cf. Pollock et al. (1989).

Usage

```
km(r, d, var = "O", conf = 0.95,age.seq=seq(1,length(r)),
ylab="Pr(survivorship from 1st age class)",xlab="Age class", ...)
```

Arguments

`r`	Numbers of individuals at risk in each age or time class.
`d`	Vector of the number of deaths in each age or time class.
`var`	Type of procedure used to calculate variance in confidence intervals `"O"` = Oakes, `"G"` = Greenwood.
`conf`	Level of confidence for confidence interval calculations; 1 - *P*(type I error)
`age.seq`	A sequence of numbers indicating the age classes used.
`ylab`	*Y*-axis label.
`xlab`	*X*-axis label.
`...`	Additional arguments from `plot`.

Details

Details for this index are given in Pollock et al. (1989).

Value

Returns a list with the following components

`s.hat`	A vector of estimated survivorship probabilities from the 1st age class onward.
`Greenwood.Var`	The estimated Greenwood variance for each age class.
`Oakes.Var`	The estimated Oakes variance for each age class.
`CI`	Upper and lower confidence bound to the true survivorship.

Author(s)

Ken Aho

References

Pollock, K. H., Winterstein, S. R., and Curtis, P. D. (1989) Survival analysis in telemetry studies: the staggered entry design. *Journal of wildlife Management*. 53(1):7-1.

Examples

```
##Example from Pollock (1989)
r<-c(18,18,18,16,16,16,15,15,13,10,8,8,7)
d<-c(0,0,2,0,0,1,0,1,1,1,0,0,0)

km(r,d)
```

`Kullback` *Kullback test for equal covariance matrices.*

Description

Provides Kullback's (1959) test for multivariate homoscedasticity.

Usage

```
Kullback(Y, X)
```

Arguments

`Y`	An n x p matrix of quantitative variables
`X`	An n x 1 vector of categorical assignments (e.g. factor levels)

Details

Multivariate general linear models assume equal covariance matrices for all factor levels or factor level combinations. Legendre and Legendre (1998) recommend this test for verifying homoscedsticiy. *P*-values evaluate a null hypothesis of equal population covariance matrices.

Value

Returns a dataframe with the test statistic (which follows a chi-square distribution if H_0 is true), the chi-square degrees of freedom, and the calculated p-value.

Author(s)

Ken Aho

References

Kullback, S. (1959) *Information theory and statistics*. John Wiley and Sons.

Legendre, P, and Legendre, L. (1998) *Numerical ecology, 2nd English edition*. Elsevier, Amsterdam, The Netherlands.

See Also

V.mat

Examples

```
Y1<-rnorm(100,10,2)
Y2<-rnorm(100,15,2)
Y<-cbind(Y1,Y2)
X<-factor(c(rep(1,50),rep(2,50)))
Kullback(Y,X)
```

KW.multi.comp *Multiple pairwise comparison procedure to accompany a Kruskal-Wallis test*

Description

As with ANOVA we can examine multiple pairwise comparisons from a Kruskal-Wallis test after we have rejected our omnibus null hypothesis. However we will need to account for the fact that these comparisons will be non-orthogonal. A conservative multiple comparison method used here is based on the Bonferroni procedure.

Usage

```
KW.multi.comp(Y, X, conf)
```

Arguments

Y	The response variable. A vector of quantitative responses.
X	An explanatory variable. A vector of factor levels.
conf	The level of desired confidence, 1 - *P*(type I error).

Value

Returns a six column dataframe containing:

1) the type of contrast (names are taken from levels in x),

2) the mean rank difference,

3) the lower confidence bound of the true mean rank difference,

4) the upper confidence bound of the true mean rank difference,

5) the hypothesis decision rule given the prescribed significance level,

6) the adjusted *p*-value.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

`Pairw.test`, `FR.multi.comp`

Examples

```
rye.data<-data.frame(rye=c(50,49.8,52.3,44.5,62.3,74.8,72.5,80.2,47.6,39.5,47.7,
50.7),nutrient=factor(c(rep(1,4),rep(2,4),rep(3,4))))
attach(rye.data)
KW.multi.comp(Y=rye,X=nutrient,conf=.95)
```

`lma.tck` *ANOVA linear models*

Description

Derives ANOVA linear model using matrix algebra

Usage

```
lma.tck()
```

Author(s)

Ken Aho

`lmr.tck` *Regression linear model derivation from linear algebra*

Description

Given Y and X matrices a regression linear model is demonstrated using matrix algebra.

Usage

```
lmr.tck()
pm1(Y, X, sz=1, showXY = TRUE)
```

Arguments

`Y`	Response variable
`X`	Explanatory variables
`sz`	Text expansion factor
`showXY`	Logical, indicating whether or not *X* and *Y* matrices should be shown.

Details

X requires a *Y* intercept variable (X0) and at least one other variable.

Author(s)

Ken Aho

See Also

`lm`

`lmu.tck` *Unbalanced and balanced linear models*

Description

The default design is ballanced, as a result Type I = Type II = Type III SS. A student can then delete one or more Y responses, and corresponding X responses to see create an unallanced design. Now the types of SS will no longer be equal. Furthermore, the order that X1 and X2 are specified will now matter in the case of Type I SS, although it will not matter for type II and III SS.

Usage

```
lmu.tck()

pm(Y, X1, X2, X1X2, change.order = FALSE)
```

Arguments

`Y`	Response variable.
`X1`	First column in design matrix with effect coding.
`X2`	Second column in design matrix.
`X1X2`	An interaction column. The product of design matrix columns one and two
`change.order`	A logical command specifying whether or not the order of X1 and X2 should changed in the model specification.

Author(s)

Ken Aho

See Also

`lm`

`loess.surf` *loess 2D and 3D smooth plots*

Description

The function serves as wrapper for `loess` and lets one make 2D or 3D smoother plots using `loess` specifications.

Usage

```
loess.surf(Y, X, span = 0.75, degree = 1, family = "gaussian", phi = 20,
theta = 50, xlab = "X", ylab = "Y", zlab = "Fit", line.col = 1,
line.type = 1, scale = TRUE, duplicate = "error", expand = 0.5, ...)
```

Arguments

`Y`	A numeric response vector.
`X`	A numeric explanatory vector or a two column matrix for 3D smooths.
`span`	Span parameter, i.e. the size of the local neighborhood.
`degree`	Indicates whether linear `degree = 1` or quadratic models `degree = 2` are to be applied to each local neighborhood.
`family`	Type of error distribution to be optimized in fitting. The default, `"gaussian"` is fitting with least squares. Fitting with Tukey's biweight *M*-Estimator is used if `family = "symmetric"`.
`phi`	Parameter from `persp`,`phi` provides the colatitude viewing angle.
`theta`	Parameter from `persp` `theta` gives the azimuthmal direction.
`xlab`	*X*-axis label.

`ylab`	*Y*-axis label.
`zlab`	*Z*-axis label
`line.col`	Color of loess fit line.
`line.type`	Line type for loess fit.
`scale`	Logical from `persp` If scale is `TRUE` the x, y and z coordinates are transformed separately. If scale is `FALSE` the coordinates are scaled so that aspect ratios are retained.
`duplicate`	Argument from `interp` from library `akima`. Consists of a character string indicating how to handle duplicate data points. The default, `duplicate = "error"` produces an error message.
`expand`	Argument from `persp`, a expansion factor applied to the z coordinates.
`...`	Additional arguments from `plot`

Value

Output is a 2D or 3D smooth plot.

Author(s)

Ken Aho

References

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`loess`

Examples

```
X1<-sort(rnorm(100))
X2<-rexp(100)
Y<-rgamma(100,1,2)
loess.surf(Y,cbind(X1,X2))
```

`loglik.plot` *Animated plots of log-likelihood functions*

Description

Plots the normal, exponential, Poisson, binomial and "custom" log-likelihood functions. Likelihoods for parameter estimates are calculated by holding data constant and varying estimates. For the normal distribution a fixed value for the parameter which is not being estimated (μ or σ^2) is established using MLEs.

Usage

```
loglik.plot(X, dist = c("norm", "poi", "bin", "exp", "custom"),
plot.likfunc = TRUE, parameter = NULL, func = NULL, poss = NULL,
plot.density = TRUE, plot.calc = FALSE, xlab = NULL, ylab = NULL,
conv = diff(range(X))/70, anim = TRUE, interval = 0.01, ...)

loglik.norm.plot(X, parameter = c("mu", "sigma.sq"), poss = NULL,
plot.likfunc = TRUE, plot.density = TRUE, plot.calc = FALSE,
xlab = NULL, ylab = NULL, conv = 0.01, anim = TRUE,
interval = 0.01, ...)

loglik.pois.plot(X, poss = NULL, plot.likfunc = TRUE,
plot.density = TRUE, plot.calc = FALSE, xlab = NULL, ylab = NULL,
conv = 0.01, anim = TRUE, interval = 0.01, ...)

loglik.binom.plot(X, poss = NULL, xlab = NULL, ylab = NULL,
plot.likfunc = TRUE, plot.density = TRUE, conv = 0.01, anim = TRUE,
interval = 0.01, ...)

loglik.exp.plot(X, poss = NULL, plot.likfunc = TRUE,
plot.density = TRUE, plot.calc = FALSE, xlab = NULL, ylab = NULL,
conv = 0.01, anim = TRUE, interval = 0.01, ...)

loglik.custom.plot(X, func, poss, anim = TRUE, interval = 0.01,
xlab, ylab, ...)

loglik.tck()
```

Arguments

`X`	A vector of quantitative data. The function does not currently handle extremely large datasets, n > 500. Data should be integers (counts) for the Poisson log-likelihood function, and binary responses (0,1) for the binomial log likelihood function. Data elements for the exponential log likelihood function must be greater than zero.
`parameter`	The parameter for which ML estimation is desired in `loglik.norm.plot` Specification of either `"mu"` or `"sigma.sq"` is required for the normal log-likelihood function. No specification is required for exponential, Poisson, and binomial log-likelihood functions since these distributions are generally specified with a single parameter, i.e. θ for the exponential, λ for the Poisson distribution, and p (the probability of a success) for the binomial distribution.
`poss`	An optional vector containing a sequence of possible parameter estimates. Elements in the vector must be distinct. If `poss` is not specified a vector of appropriate possibilities is provided by the function. This argument can be used to set `xlim` in the likelihood function and density plots.
`dist`	The type of assumed distribution there are currently five possibilities: `"norm"`, `"poi"`, `"binom"`, `"exp"`, and `"custom"`. Use of custom distributions requires specification of a custom likelihood function in the argument `func`.

`plot.likfunc`	A logical command for indicating whether a graph of the log-likelihood function should be created.
`plot.density`	A logical command for indicating whether a second graph, in which densities are plotted on the pdf, should be created.
`plot.calc`	A logical command for indicating whether a third graph, in which log-densities are added to one another, should be created.
`xlab`	Optional X-axis label.
`ylab`	Optional Y-axis label.
`conv`	Precision of likelihood function. Decreasing `conv` increases the smoothness and precision of the ML function. Decreasing `conv` will also slow the animation.
`anim`	A logical command indicating whether animation should be used in plots.
`interval`	Speed of animation, in seconds per frame. May not work in all systems; see `Sys.sleep`.
`func`	Custom likelihood function to be specified when using `loglik.custom.plot`. The function should have two arguments. An optional call to data, and the likelihood function parameter (see example below).
`...`	Additional arguments from `plot` can be specified for likelihood function plots.

Details

These plots are helpful in explaining the workings of ML estimation for parameters. Animation is included as an option to further clarify processes. When specifying `poss` be sure to include the estimate that you "want" the log-likelihood function to maximize in the vector of possibilities, e.g. `mean(X)` for estimation of μ.

Value

Three animated plots can be created simultaneously. The first plot shows the normal, Poisson, exponential, binomial, or custom log-likelihood functions. The second plot shows the pdf with ML estimates for parameters. On this graph densities of observations are plotted as pdf parameters are varied. By default these two graphs will be created simultaneously on a single graphics device. By specifying `plot.calc = TRUE` a third plot can also be created which shows that log-likelihood is the sum of the log-densities. Animation in this third plot will be automatically sped up, using a primitive routine, for large datasets, and slowed for small datasets. The third plot will not be created for the binomial pdf becasue there will only be a single outcome from the perspective of likelihood (e.g. 10 successes out of 22 trials). The second and third plots will not be created for custom likelihood functions. Loading package **tcltk** allows use of the function `loglik.tck` which provides an interactive GUI to run `loglik.plot`.

Author(s)

Ken Aho

See Also

`dnorm`, `dpois`, `dexp`, `dbinom`

Examples

```
## Not run:
##Normal log likelihood estimation of mu.
X<-c(11.2,10.8,9.0,12.4,12.1,10.3,10.4,10.6,9.3,11.8)
loglik.plot(X,dist="norm",parameter="mu")

##Add a plot describing log-likelihood calculation.
loglik.plot(X,dist="norm",parameter="mu",plot.calc=TRUE)

##Normal log likelihood estimation of sigma squared.
X<-c(11.2,10.8,9.0,12.4,12.1,10.3,10.4,10.6,9.3,11.8)
loglik.plot(X,dist="norm",parameter="sigma.sq")

##Exponential log likelihood estimation of theta
X<-c(0.82,0.32,0.14,0.41,0.09,0.32,0.74,4.17,0.36,1.80,0.74,0.07,0.45,2.33,0.21,
0.79,0.29,0.75,3.45)
loglik.plot(X,dist="exp")

##Poisson log likelihood estimation of lambda.
X<-c(1,3,4,0,2,3,4,3,5)
loglik.plot(X,dist="poi")

##Binomial log likelihood estimation of p.
X<-c(1,1,0,0,0,1,0,0,0,0)#where 1 = a success
loglik.plot(X,dist="bin",interval=.2)

##Custom log-likelihood function
func<-function(X=NULL,theta)theta^5*(1-theta)^10
loglik.plot(X=NULL,func=func,dist="custom",poss=seq(0,1,0.01),
xlab="Possibilities",ylab="Log-likelihood")

##Interactive GUI, requires package 'tcltk'
loglik.tck()

## End(Not run)
```

---

magnets *Magnet pain relief data*

Description

Magnets have long been used as an alternative medicine, particularly in the Far East, for speeding the recovery of broken bones and to aid in pain relief. Valbona et al. (1997)tested whether chronic pain experienced by post-polio patients could be treated with magnetic fields applied directly to pain trigger points. The investigators identified fifty subjects who not only had post-polio syndrome, but who also experienced muscular or arthritic pain. Magnets were applied to pain trigger points in 29 randomly selected subjects, and in the other 21 a placebo was applied. The patients were asked to subjectively rate pain on a scale from one to ten before and after application of the magnet or placebo.

Usage

```
data(magnets)
```

Format

The dataframe contains 4 columns

Score_1 Reported pain level before application of treatment.

Score_2 Reported pain level after application of treatment.

Active Categorical variable indicating whether the device applied was active (magent) or inactive (placebo).

Source

Vallbona, C. et al. 1997. Response of pain to static magnetic fields in postpolio patients, a double blind pilot study. *Archives of Physical Medicine and Rehabilitation*. 78: 1200-1203.

`MC.test` *Monte Carlo hypothesis testing for two samples.*

Description

`MC.test2` calculates a permutation of test statistics from an pooled variance *t*-test. It compares this distribution to an initial test statistic calculated using non-permuted data to derive a *p*-value.

Usage

```
MC.test(Y,X, perm = 1000, alternative = c("less", "greater", "not.equal"))
```

Arguments

`Y`	Response data.
`X`	Categorical explanatory variable.
`perm`	Number of iterations.
`alternative`	Alternative hypothesis. One of three options: `"less"`,`"greater"`, or `"not.equal"`. These provide lower-tail, upper-tail, and two-tailed tests.

Details

The method follows the description of Manly (1998) for a two-sample test. The pooled variance *t*-test procedure assumes homoscedasticity for the populations being compared. Upper and lower tailed tests are performed by finding the portion of the distribution greater than or equal to the observed test statistic (upper-tailed) or less than or equal to the observed test statistic (lower-tailed). A two tailed test is performed by multiplying the portion of the null distribution above the absolute value of the observed test statistic by two. Results from the test will be similar to `oneway_test` from the library `coin` since it is based on an equivalent test statistic. The function `oneway_test` allows additional options including blocking.

Value

Returns a list with the following items:

`observed.test.statistic`
: t-statistic calculated from non-permuted (original)data.

`no_of_permutations_exceeding_observed_value`
: The number of times a Monte Carlo derived test statistic was more extreme than the initial observed test statistic.

`p.value`
: Empirical p-value

`alternative`
: The alternative hypothesis

Author(s)

Ken Aho

References

Manly, B. F. J. (1997) *Randomization and Monte Carlo methods in biology, 2nd edition*. Chapman and Hall, London.

See Also

`t.test`

Examples

```
Y<-c(runif(100,1,3),runif(100,1.2,3.2))
X<-factor(c(rep(1,100),rep(2,100)))
MC.test(Y,X,alternative="less")
```

Mode *Sample mode*

Description

Calculates the sample mode; i.e. the most frequent outcome in a dataset. Non-existence of the mode will return a message.

Usage

```
Mode(x)
```

Arguments

`x`	A vector of quantitative data.

Value

Returns the sample mode or an error message if the mode does not exist.

Author(s)

Ken Aho

References

Bain, L. J., and M. Engelhardt (1992) *Introduction to probability and mathematical statistics*. Duxbury press. Belmont, CA, USA.

See Also

H.mean, HL.mean, mean, median, huber.mu

Examples

```
x<-round(rnorm(100000,mean=10,sd=2),0)
mode(x)
```

---

modlevenes.test *Modified Levene's test*

Description

Conducts the modified Levene's test for homoscedastic populations.

Usage

```
modlevenes.test(x, groups)
```

Arguments

x	Vector of residuals from a linear model.
groups	Vector of factor levels.

Details

The modified Levene's test is a test for homoscedsticity that (unlike the classic *F*-test) is robust to violations of normality (Conover et al. 1981). In a Modified Levene's test we calculate $d_{ij} = |e_{ij} - \tilde{e}_i|$ where $\tilde{e}_i$ is the *i*th factor level residual median. We then run an ANOVA on the d_{ij}'s. If the *p*-value is < α, we reject the null and conclude that the population error variances are not equal.

Value

An ANOVA table is returned with the modified Levene's test results.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

fligner.test

Examples

```
eggs<-c(11,17,16,14,15,12,10,15,19,11,23,20,18,17,27,33,22,26,28)
trt<-as.factor(c(1,1,1,1,1,2,2,2,2,2,3,3,3,3,4,4,4,4,4))
lm1<-lm(eggs~trt)
modlevenes.test(residuals(lm1),trt)
```

Mrpp — *Multi-response permutation procedure. A wrapper for mrpp from vegan.*

Description

While mrpp in library vegan directly tests the hypothesis of identical multivariate distributions using a permutation procedure, PC-ORD approximates the distribution of using a Pearson type III distribution which accommodates the fact that the permutation distribution is often significantly skewed (McCune and Grace 2002). The Pearson type III distribution is similar to the binomial distribution when $p \neq q$ (Abramowitz and Stegun 1972).

Usage

```
Mrpp(dat, grouping, permutations = 1000, distance = "euclidean",
weight.type = 1, strata)
```

Arguments

dat	data matrix or data frame in which rows are samples and columns are response variable(s), or a dissimilarity object or a symmetric square matrix of dissimilarities.
grouping	Factor or numeric index for grouping observations.
permutations	Factor or numeric index for grouping observations.
distance	Choice of distance metric that measures the dissimilarity between two observations . The function vegdist is used to calculate dissimilarities.

`weight.type`	choice of group weights. See Details below for options.
`strata`	An integer vector or factor specifying the strata for permutation. If supplied, observations are permuted only within the specified strata.

Details

`Mrpp` provides additional output to `mrpp` including a T statistic and a Pearson's type III p-value. The Pearson Type III distribution has three parameters: a mean, standard deviation , and a shape parameter , based on the skewness of the empirical distribution. These are estimated, and p-values are calculated in the wrapper function `Mrpp` using functions from the library `lmomco` (Asquith 2008). See documentation from `mrpp` for additional information.

Value

The function returns a list of class `Mrpp` with following items:

`call`	Function call.
`delta`	The overall weighted mean of group mean distances.
`E.delta`	expected delta, under the null hypothesis of no group structure. This is the mean of original dissimilarities.
`CS`	Classification strength (Van Sickle 1997) with `weight.type = 3` and `NA` with other weights.
`n`	Number of observations in each class.
`classdelta`	Mean dissimilarities within classes. The overall delta is the weighted average of these values with given `weight.type`.
`Pvalue`	Significance of the test.
`distance`	Choice of distance metric used; the "method" entry of the dist object.
`weight.type`	The choice of group weights used.
`boot.deltas`	The vector of "permuted deltas," the deltas calculated from each of the permuted datasets.
`permutations`	The number of permutations used.

Author(s)

M. Herny H. Stevens `<HStevens@muohio.edu>` and Jari Oksanen. Wrapper created by Ken Aho

References

McCune, B. and Grace, J. B. (2002) *Analysis of Ecological Communities. MjM Software Design*, Gleneden Beach, Oregon, USA.

Mielke P. W. and Berry, K. J. (2001) *Permutation Methods: A Distance Function Approach*. Springer Series in Statistics. Springer.

Van Sickle, J. (1997) Using mean similarity dendrograms to evaluate classifications. *Journal of Agricultural, Biological, and Environmental Statistics* 2:370-388.

Examples

```
## Not run:
library(vegan)
data(dune)
data(dune.env)
Mrpp(dune, dune.env$Management)
## End(Not run)
```

`MS.test` *Mack-Skillings test*

Description

Runs a Mack-Skillings test for situations applicable to rank-based permutation procedures with blocking and more than one replicate for treatments in a block.

Usage

```
MS.test(Y, X, reps)
```

Arguments

`Y`	A matrix of response data. The `MS.test` function requires that response data are organized in columns (see example below).
`X`	A vector of treatments. The length of the vector should be equal to the number of rows in the response matrix.
`reps`	The number of replicates in each treatment (unbalanced designs cannot be analyzed).

Details

When we have more than one replication within a block, and the number of replications is equal for all treatments, we can use the Mack-Skillings test (Mack and Skillings 1980) as a rank based permutation procedure to test for main effect differences. If ties occur the value of the significance level is only approximate. Hollander and Wolfe (1996) provide a method for finding exact *p*-values by deriving a test statistic distribution allowing ties.

Value

Returns a dataframe summarizing the degrees of freedom, test statistic and *p*-value.

Author(s)

Ken Aho

References

Campbell, J. A., and O. Pelletier (1962) Determination of niacin (niacinamide) in cereal products. *J. Assoc. Offic. Anal. Chem.* 45: 449-453.

Hollander, M., and D. A. Wolfe (1999) *Nonparametric statistical methods*. New York: John Wiley & Sons.

Mack, G. A., and J. H. Skillings (1980) A Friedman-type rank test for main effects in a two-factor ANOVA. *Journal of the American Statistical Association*. 75: 947-951.

See Also

friedman.test

Examples

```
#data from Campbell and Pelletier (1962)
Niacin0<-c(7.58,7.87,7.71,8.00,8.27,8,7.6,7.3,7.82,8.03,7.35,7.66)
Niacin4<-c(11.63,11.87,11.40,12.20,11.70,11.80,11.04,11.50,11.49,11.50,10.10,11.70)
Niacin8<-c(15.00,15.92,15.58,16.60,16.40,15.90,15.87,15.91,16.28,15.10,14.80,15.70)
Niacin<-cbind(Niacin0,Niacin4,Niacin8)
lab<-c(rep(1,3),rep(2,3),rep(3,3),rep(4,3))
MS.test(Niacin, lab, reps=3)
```

---

near.bound — *Nearest neighbor boundary coordinates*

Description

Finds nearest neighbor boundary Cartesian coordinates for use as arguments in function prp.

Usage

```
near.bound(X, Y, bX, bY)
```

Arguments

X	A vector of Cartesian *X*-coordinates (e.g. UTMs) describing an animal's locations (e.g. telemetry data).
Y	A vector of Cartesian *Y* coordinates (e.g. UTMs) describing an animal's locations (e.g. telemetry data).
bX	A vector of boundary *X*-coordinates.
bY	A vector of boundary *Y*-coordinates.

Value

Returns Cartesian *X,Y* coordinates of nearest neighbor locations on a boundary.

Author(s)

Ken Aho

See Also

prp, bound.angle

Examples

```
bX<-seq(0,49)/46
bY<-c(4.89000,4.88200,4.87400,4.87300,4.88000,4.87900,4.87900,4.90100,4.90800,
4.91000,4.93300,4.94000,4.91100,4.90000,4.91700,4.93000,4.93500,4.93700,
4.93300,4.94500,4.95900,4.95400,4.95100,4.95800,4.95810,4.95811,4.95810,
4.96100,4.96200,4.96300,4.96500,4.96500,4.96600,4.96700,4.96540,4.96400,
4.97600,4.97900,4.98000,4.98000,4.98100,4.97900,4.98000,4.97800,4.97600,
4.97700,4.97400,4.97300,4.97100,4.97000)

X<-c(0.004166667,0.108333333,0.316666667,0.525000000,0.483333333,0.608333333,
0.662500000,0.683333333,0.900000000,1.070833333)
Y<-c(4.67,4.25,4.26,4.50,4.90,4.10,4.70,4.40,4.20,4.30)
near.bound(X,Y,bX,bY)
```

---

one.sample.t *One sample t-test*

Description

Provides a one-sample hypothesis test. The test assumes that the underlying population is normal.

Usage

```
one.sample.t(data = NULL, null.mu = 0, xbar = NULL, s, n = NULL,
test = c("two.sided", "lower.tail", "upper.tail"))
```

Arguments

data	A vector of quantitative data. Not required if xbar and n are supplied by the user.
null.mu	The expectation for the null distribution.
xbar	Sample mean. Not required if is.null(data)==FALSE
s	The sample standard deviation. Not required if is.null(data)==FALSE
n	The sample size. Not required if is.null(data)==FALSE
test	Type of test. One of three must be specified "two.sided", "lower.tail", or "upper.tail"

Details

The function can use either raw data `is.null(data)==FALSE` or summarized data if `is.null(data)==TRUE`. With the later `xbars`, and `n` must be specified by the user.

Value

Returns a test statistic and a *p*-value.

Author(s)

Ken Aho

See Also

`pt`

Examples

```
one.sample.z(null.mu=131,xbar=126,s=12,n=85,test="two.sided")
```

`one.sample.z` *One sample z-test*

Description

Provides a one-sample hypothesis test. The test assumes that the underlying population is normal and furthermore that σ is known.

Usage

```
one.sample.z(data = NULL, null.mu = 0, xbar = NULL, sigma, n = NULL,
test = c("two.sided", "lower.tail", "upper.tail"))
```

Arguments

`data`	A vector of quantitative data. Not required if `xbar` and `n` are supplied by the user.
`null.mu`	The expectation for the null distribution.
`xbar`	Sample mean. Not required if `is.null(data)==FALSE`
`sigma`	The null distribution standard deviation
`n`	The sample size. Not required if `is.null(data)==FALSE`
`test`	Type of test. One of three must be specified `"two.sided", "lower.tail"`, or `"upper.tail"`

Details

The function can use either raw data `is.null(data)==FALSE` or summarized data if `is.null(data)==TRUE`. With the later `xbar` and `n` must be specified by the user.

Value

Returns a test statistic and a p-value.

Author(s)

Ken Aho

See Also

`pnorm`

Examples

```
one.sample.z(null.mu=131,xbar=126,sigma=12,n=85,test="two.sided")
```

`Pairw.test` *Conducts pairwise post hoc tests associated with an ANOVA.*

Description

Conducts all possible pairwise tests with adjustments to p-values using one of four methods: Least Significant difference, Bonferroni, Tukey's honest significantly difference (HSD), or Scheffe's method.

Usage

```
Pairw.test(y, x, conf.level = 0.95, method = c("LSD", "Bonf", "Tukey",
"Sheffe"), MSE = NULL, df.err = NULL)

LSD.test(y, x, conf.level = 0.95, MSE = NULL, df.err = NULL)

BonferroniCI(y, x, conf.level = 0.95, MSE = NULL, df.err = NULL)

TukeyCI(y, x, conf.level = 0.95, MSE = NULL, df.err = NULL)

ScheffeCI(y, x, conf.level = 0.95, MSE = NULL, df.err = NULL)
```

Arguments

y	A quantitative vector containing the response variable
x	A categorical vector containing the groups (e.g. factor levels or treatments)
conf.level	1 - *P*(type I error)
method	One of four possible choices: "LSD", "Bonf", "Tukey", "Sheffe"
MSE	Value of MSE from the omnibus ANOVA. Default = NULL
df.err	Degrees of freedom error from the omnibus ANOVA. Default = NULL

Details

Adjustment for simultaneous inference is a contentious subject and will not be discussed at length here. For description of methods go to Kutner et al. (2005). For multifactor models, MSE and the residual degrees of freedom (used in the computation of confidence intervals for all pairwise methods used here) will vary depending on the experimental design and the number of factors. Thus, for multifactor designs the user should specify the residual degrees of freedom and MSE from the omnibus ANOVA. This will be unnecessary for one-way ANOVAs

Value

Returns a six column dataframe containing:

1) the type of contrast (names are taken from levels in x),

2) the mean difference,

3) the lower confidence bound of the true mean difference,

4) the upper confidence bound of the true mean difference,

5) the hypothesis decision rule given the prescribed significance level, and

6) the adjusted *p*-value.

Note

Different forms of these functions have existed for years without implementation into libraries. My version here, based on the function outer is unique.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and Li., W (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

TukeyHSD

Examples

```
eggs<-c(11,17,16,14,15,12,10,15,19,11,23,20,18,17,27,33,22,26,28)
trt<-as.factor(c(1,1,1,1,1,2,2,2,2,2,3,3,3,3,4,4,4,4,4))
Pairw.test(x=trt,y=eggs,method="LSD")##LSD method
Pairw.test(x=trt,y=eggs,method="Bonf")##Bonferroni
Pairw.test(x=trt,y=eggs,method="Tukey")##Tukey HSD
Pairw.test(x=trt,y=eggs,method="Scheffe")##Sheffe
```

`panel.cor.res` *functions for customizing correlation matrices*

Description

The functions here can be used to customize upper and lower triangles in correlation matrices. In particular `panel.cor.res` provides correlation coefficients (any alternative from `cor` can be used) and *p*-values for correlation tests. The function `panel.lm` puts linear fitted lines from simple linear regression in scatterplots. Note that the function `panel.smooth` provides a smoother fit.

Usage

```
panel.cor.res(x, y, digits = 2, meth = "pearson", cex.cor=1)
panel.lm(x, y, col = par("col"), bg = NA, pch = par("pch"), cex = 1,
col.line = 2,lty = par("lty"))
```

Arguments

`x`	variable 1 in correlation
`y`	variable 2 in correlation
`digits`	number of digits in text for `panel.cor.res`
`meth`	type of correlation coefficeint from panel.cor.res, one of `"pearson"`, `"spearman"`, `"kendall"`
`cex.cor`	size of text in `panel.lm`
`col`	color of points in `panel.lm`
`bg`	background color of points in `panel.lm`
`pch`	type of symbols for points in `panel.lm`
`cex`	symbol size in `panel.lm`
`lty`	line type in `panel.lm`
`col.line`	color of lines in `panel.lm`

Author(s)

Ken Aho

See Also

cor, cor.test, panel.smooth

Examples

```
data(asthma)
attach(asthma)
pairs(asthma,cex.labels=1,cex=.95,gap=.1,lower.panel=panel.cor.res,
upper.panel=panel.lm)
```

`partial.R2` *Partial correlations of determination in multiple regression*

Description

Calculates the partial correlation of determination for a variable of interest in a multiple regression.

Usage

```
partial.R2(lm.without, lm.with)
```

Arguments

`lm.without`	A linear model without the variable of interest.
`lm.with`	A linear model with the variable of interest.

Details

Coefficients of partial determination measure the proportional reduction in sums of squares after a variable of interest, *X*, is introduced into a model. We can see how this would be of interest in a multiple regression.

Value

The partial r^2 is returned.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

cor, partial.resid.plot

Examples

```
Soil.C<-c(13,20,10,11,2,25,30,25,23)
Soil.N<-c(1.2,2,1.5,1,0.3,2,3,2.7,2.5)
Slope<-c(15,14,16,12,10,18,25,24,20)
Aspect<-c(45,120,100,56,5,20,5,15,15)
Y<-as.vector(c(20,30,10,15,5,45,60,55,45))

lm.with<-lm(Y~Soil.C+Soil.N+Slope+Aspect)
lm.without<-lm(Y~Soil.C+Slope+Aspect)

partial.R2(lm.without,lm.with)
```

---

partial.resid.plot *Partial residual plots for interpretation of multiple regression.*

Description

The function creates partial residual plots which help a user graphically determine the effect of a single predictor with respect to all other predictors in a multiple regression model.

Usage

```
partial.resid.plot(Y, X, smooth.span = 0.8)
```

Arguments

Y	A vector of quantitative responses.
X	A matrix of explanatory variables.
smooth.span	Degree of smoothing for smoothing line.

Details

Creates partial residual plots (see Kutner et al. 2002). Smoother lines from lowess and linear fits from lm are imposed over plots to help an investigator determine the effect of a particular *X* variable on *Y* with all other variables in the model. The function automatically inserts explanatory variable names on axes.

Value

Returns p partial residual plots, where p = the number of explanatory variables.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

See Also

`partial.R2`

Examples

```
Soil.C<-c(13,20,10,11,2,25,30,25,23)
Soil.N<-c(1.2,2,1.5,1,0.3,2,3,2.7,2.5)
Slope<-c(15,14,16,12,10,18,25,24,20)
Aspect<-c(45,120,100,56,5,20,5,15,15)
Y<-as.vector(c(20,30,10,15,5,45,60,55,45))
X<-as.matrix(cbind(Soil.C, Soil.N, Slope, Aspect))
par(mfrow=c(2,2),mar=c(5,4,1,1.5))
partial.resid.plot(Y,X)
```

`perm.fact.test` *Permutation test for two and three way factorial designs.*

Description

Provides permutation tests for two and three way designs, using permutations of of the response vector with respect to factor levels. One way permutation tests are provided by `MC.test`, and the function `oneway_test` in `coin`.

Usage

```
perm.fact.test(Y, X1, X2, X3 = NA, perm = 100)
```

Arguments

`Y`	A vector of response data. A quantitative vector.
`X1`	A vector of factor levels describing factor one.
`X2`	A vector of factor levels describing factor two.
`X3`	If necessary, a vector of factor levels describing factor three.
`perm`	Number of permutations.

Details

The method used here is based on Manly (1997).

Value

A dataframe is returned describing initial F test statistics for main effects and interactions, degrees of freedom, and permutation p-values.

Author(s)

Ken Aho

References

Manly, B. F. J. (1997) *Randomization and Monte Carlo methods in biology, 2nd edition*. Chapman and Hall, London.

See Also

MC.test

Examples

```
lizard<-data.frame(ants=c(13,242,105,8,59,20,515,488,88,18,44,21,182,21,7,24,312,68,
460,1223,990,140,40,27),size=factor(c(rep(1,12),rep(2,12))),
month=factor(rep(rep(c(1,2,3,4),each=3),2)))
attach(lizard)
perm.fact.test(ants,month,size,perm=100)
```

---

plantTraits *Plant traits for 136 species*

Description

This dataset, from the library `cluster`, describes 136 plant species according to biological attributes (morphological or reproductive).

Usage

```
data(plantTraits)
```

Format

A data frame with 136 observations on the following 31 variables.

pdias Diaspore mass (mg).

longindex Seed bank longevity.

durflow Flowering duration.

height Plant height, an ordered factor with levels '1' < '2' < ... < '8'.

begflow Time of first flowering, an ordered factor with levels '1' < '2' < '3' < '4' < '5' < '6' < '7' < '8' < '9'.

mycor Mycorrhizas, an ordered factor with levels '0'never < '1' sometimes< '2'always.

vegaer Aerial vegetative propagation, an ordered factor with levels '0'never < '1' present but limited< '2'important.

vegsout Underground vegetative propagation, an ordered factor with 3 levels identical to 'vegaer' above.

autopoll Selfing pollination, an ordered factor with levels '0'never < '1'rare < '2' often< the rule'3'.

insects Insect pollination, an ordered factor with 5 levels '0' < ... < '4'.

wind Wind pollination, an ordered factor with 5 levels '0' < ... < '4'.

lign A binary factor with levels '0:1', indicating if plant is woody.

piq A binary factor indicating if plant is thorny.

ros A binary factor indicating if plant is rosette.

semiros Semi-rosette plant, a binary factor ('0': no; '1': yes).

leafy Leafy plant, a binary factor.

suman Summer annual, a binary factor.

winan Winter annual, a binary factor.

monocarp Monocarpic perennial, a binary factor.

polycarp Polycarpic perennial, a binary factor.

seasaes Seasonal aestival leaves, a binary factor.

seashiv Seasonal hibernal leaves, a binary factor.

seasver Seasonal vernal leaves, a binary factor.

everalw Leaves always evergreen, a binary factor.

everparti Leaves partially evergreen, a binary factor.

elaio Fruits with an elaiosome (dispersed by ants), a binary factor.

endozoo Endozoochorous fruits, a binary factor.

epizoo Epizoochorous fruits, a binary factor.

aquat Aquatic dispersal fruits, a binary factor.

windgl wind dispersed fruits, a binary factor.

unsp Unspecialized mechanism of seed dispersal, a binary factor.

Details

Most of factor attributes are not disjunctive. For example, a plant can be usually pollinated by insects but sometimes self-pollination can occur.

Note

The description here follows directly from that in `cluster`.

Source

Vallet, Jeanne (2005) *Structuration de communautes vegetales et analyse comparative de traits biologiques le long d'un gradient d'urbanisation*. Memoire de Master 2 'Ecologie-Biodiversite-Evolution'; Universite Paris Sud XI, 30p.+ annexes (in french).

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M. (2005). *Cluster Analysis Basics and Extensions*; unpublished.

`plotCI.reg` *Plots a simple linear regression along with confidence and prediction intervals.*

Description

Plots the fitted line from a simple linear regression (y ~ x) and (if requested) confidence and prediction intervals.

Usage

```
plotCI.reg(x, y, conf = 0.95, CI = TRUE, PI = TRUE, reg.col = 1, CI.col = 2,
PI.col = 4, reg.lty = 1, CI.lty = 2, PI.lty = 3, reg.lwd = 1, CI.lwd = 1, ...)
```

Arguments

`x`	The explanatory variable, a numeric vector.
`y`	The response variable, a numeric vector
`conf`	The level of confidence; 1 - *P*(type I error)
`CI`	Logical; should the confidence interval be plotted?
`PI`	Logical; should the prediction interval be plotted?
`reg.col`	Color of the fitted regression line.
`CI.col`	Color of the confidence interval lines.
`PI.col`	Color of the prediction interval lines.
`reg.lty`	Line type for the fitted regression line.
`CI.lty`	Line type for the confidence interval.
`PI.lty`	Line type for the confidence interval.
`reg.lwd`	Line width for the regression line.
`CI.lwd`	Line widths for the confidence and prediction intervals.
`...`	Additional arguments from `plot`

Value

Returns a plot with a regression line and (if requested) confidence and prediction intervals

Author(s)

Ken Aho

See Also

plot, predict

Examples

```
y<-c(1,2,1,3,4,2,3,4,3,5,6)
x<-c(2,3,1,4,5,4,5,6,7,6,8)
plotCI.reg(x,y)
```

---

polar.ord *Polar ordinations*

Description

The function currently creates two or three dimensional Bray-Curtis (polar) ordinations.

Usage

```
polar.ord(data, index = "steinhaus", endpoint = c("BC.original",
"PC_ORD.original", "var.reg"), get.resid.dist = FALSE)
```

Arguments

data	A multivariate dataset, e.g. a community site x species matrix.
index	The dissimilarity or distance measure to be used. Any method from get.dist can be used.
endpoint	Endpoint selection is accomplished using either the Bray-Curtis original "BC.original" method (i.e. using the sites which are furthest apart), the PC-ORD original method "PC_ORD.original" (the first endpoint has the highest sum of distances from other sites, the second endpoint has the highest distance from the first endpoint), or using the variance regression "var.reg" method as described by McCune and Grace (2002). Note that the variance regression method in PC-ORD does not appear to be doing what it is supposed to be doing. The variance regression method in polar.ord is in accordance with in McCune and Grace (2002).
get.resid.dist	Logical. Allows one to see the residual distance matrices.

Details

The Bray-Curtis method is a relatively easy to understand ordination procedure that is also known as polar ordination because it arranges points in reference to endpoints or poles. The method is strongly favored by ecologists from the University of Wisconsin (probably because this is the school John Curtis graduated from and taught at). Curtis developed this method with the Canadian ecologist James Bray in the early 50s.

While polar ordination seems to work reasonably well for recovering community patterns, it has recently been very difficult to publish papers using this technique for two reasons: 1) it has been judged "outmoded" compared to more recently developed matrix decomposition methods, i.e. DCA and CCA, and 2) the endpoint selection techniques for axes can be arbitrary (more on this later). Edward Beals (1984) wrote a scathing critique of ecologists who undersold polar ordination. Here is an excerpt:

"While ordination as an approach to data analysis gained acceptance in the 1960's, the Bray-Curtis method came under attack beginning with Austin and Orloci (1966), and it quickly fell into disfavor among ecologists as new methods of ordination were introduced and championed. Only ecologists trained at the University of Wisconsin persisted in using Bray-Curtis, not out of blind loyalty, but because it generally gave more ecologically interpretable results than did newer or more sophisticated methods"

Endpoint selection is the most crucial step in the polar ordination process, because all other points will be placed in relation to the endpoints. The original Bray-Curtis method used the two most dissimilar points as endpoints. Two other methods, `"PC_ORD.original"` and `"var.reg"` are also allowed by `polar.ord`.

Value

Output includes scores, the amount of variance explained by axes, and, if requested, the residual distance matrices.

Author(s)

Ken Aho

References

Beals, E. W. 1984. Bray-Curtis ordination: an effective strategy for analysis of multivariate ecological data. *Advances in Ecological Research*. 14: 1-55.

McCune, B., and J.B. Grace. 2002. *Analysis of ecological communities*. MjM Software design. Gelenden Beach OR.

McCune, B. and E. W. Beals. 1993. History of the development Bray-Curtis ordination. Pp. 67-72 in J. S. Fralish, R. P. McIntosh, and O. L. Loucks eds. *John Curtis: Fifty Years of Wisconsin Plant Ecology*. Wisconsin Academy of Science, Arts and Letters, Madison WI.

See Also

`get.dist`

Examples

```
demodat<-matrix(ncol=3,nrow=5,data=c(2,3,5,7,9,1,4,0,6,2,5,5,10,2,2),
byrow=FALSE)
po.orig <-polar.ord(demodat,endpoint="BC.original",get.resid.dist=FALSE)
```

`power.z.test` *Power analysis for a one sample z-test*

Description

A power analysis for a one sample z-test. The function requires alpha, sigma, the effect size, the type of test (one tailed or two-tailed), and either power (1 - beta) or *n* (sample size). If *n* is provided, then power is calculated. Conversely, if one provides power, but not *n*, then n is calculated.

Usage

```
power.z.test(sigma = 1, n = NULL, power = NULL, alpha = 0.05, effect = NULL,
test = c("two.tail", "one.tail"), strict = FALSE)
```

Arguments

`sigma`	The population variance
`n`	The sample size. Not required if `power` is specified.
`power`	The desired power. Not required if `n` is specified.
`alpha`	Probability of type I error.
`effect`	Effect size.
`test`	One of two choices: `"two.tail"` or `"one.tail"`.
`strict`	Causes the function to use a strict interpretation of power in a two-sided test. If strict = TRUE then power for a two sided test will include the probability of rejection in the opposite tail of the true effect. If strict = FALSE (the default) power will be half the value of α if the true effect size is zero.

Value

Returns a list

`sigma`	The prescribed population variance.
`n`	The sample size.
`power`	The power.
`alpha`	The type I error probability.
`test`	The type of test prescribed.
`effect`	The effect size.

Author(s)

Ken Aho

References

Bain, L. J., and M. Engelhardt (1992) *Introduction to probability and mathematical statistics*. Duxbury press. Belmont, CA, USA.

See Also

pnorm

Examples

```
power.z.test(sigma=6,effect=5,power=.9,test="one.tail")
```

`press` *prediction sum of squares*

Description

Calculates prediction sum of squares (PRESS) for a linear model.

Usage

```
press(Y, X)
```

Arguments

Y	A numeric vector containing the response variable.
X	A vector or matrix containing explanatory variables.

Details

The press statistic is calculated as:

$$\sum_{i=1}^{n} d_i^2$$

where

$$d_i = \frac{e_i}{1 - h_{ii}}$$

where h_{ii} is the *i*th diagonal element in the hat matrix.

Value

Returns the PRESS statistic.

Author(s)

Ken Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. 2005. *Applied linear statistical models, 5th edition.* McGraw-Hill, Boston.

See Also

`cor`

Examples

```
Y<-rnorm(100)
X<-rnorm(100)
press(Y,X)
```

Preston.dist — *Preston diversity analysis*

Description

A diversity and richness analysis method based on the Preston (1948) log-normal distribution.

Usage

```
Preston.dist(counts, start = 0.2)
```

Arguments

`counts`	Vector of counts for species in a community dataset.
`start`	Starting value for non-linear least squares estimation of *a* in $n = n_0 \times e^{-aR^2}$.

Details

Preston (1948) proposed that after a log2 transformation species abundances, grouped in bins representing a doubling of abundance (octaves), would be normally distributed. Thus, after this transformation most species in a sample would have intermediate abundance, and there would be relatively few rare or ubiquitous species. The Preston model is based on the Gaussian function: $n = n_0 \times e^{-aR^2}$, where, *n*0 is the number of species contained in the modal octave, *n* is the number of species contained in an octave *R* octaves from the modal octave, and *a* is an unknown parameter. The parameter *a* is estimated using the function `nls`, using a starting value, 0.2, recommended by Preston. The area under Preston curve provides an extrapolated estimate of richness and thus an indication of the adequacy of a sampling effort. Preston called a line placed at the 0th octave the veil line. He argued that species with abundances below the veil line have not been detected due to inadequate sampling.

Value

Graph of the Preston log-normal distribution for a dataset given by "counts", and a summary of the analysis including the fitted Gaussian equation, the estimated number of species, and an estimate for the percentage of sampling that was completed i.e. `length(counts)/Est.no.of.spp*100`.

Author(s)

Ken Aho

References

Preston, F.W. (1948) The commonness and rarity of species. *Ecology* 29, 254-283.

See Also

`dnorm`, `nls`

Examples

```
library(vegan)
data(BCI)
BCI.count<-apply(BCI,2,sum)
Preston.dist(BCI.count)
```

---

prp *Perpendicularity*

Description

Calculates a perpendicularity index, η, for animal spatial movements. The index has a [0, 1] range with 0 indicating a perfectly parallel movement with respect to boundary or edge and 1 indicating perfectly perpendicular movement. Other summaries are also provided.

Usage

```
prp(Time, S.X, S.Y, N.X, N.Y, habitat = NULL, near.angle = NULL,
F.0.NA = TRUE)
```

Arguments

`Time`	A numeric vector containing the times when spatial coordinates were recorded.
`S.X`	*X*-coordinates of animal.
`S.Y`	*Y*-coordinates of animal.
`N.X`	*X*-coordinate of nearest point on boundary. These data can be obtained from function `near.bound` or from ARCGIS Near output.

`N.Y` Y-coordinate of nearest point on boundary. These data can be obtained from function `near.bound` or from ARCGIS Near output.

`habitat` A character vector of habitat categories.

`near.angle` A numeric vector containing the angle of azimuth to the nearest point on the boundary with respect to a four quadrant system. NE = $0°$ to $90°$, NW is > $90°$ and $\leq 180°$, SE is < $0°$ and $\leq -90°$ is > $-90°$ and $\leq -180°$. This output can be obtained from function `bound.angle` or from ARCGIS Near output.

`F.0.NA` A logical argument specifying whether or not a time interval in which F = 0 should be made `NA` (see Figure from examples)

Details

This index for perpendicularity, η is based on the following rules:

if $\delta \leq 90^o$ then η = $\delta/90^o$; if $90^o < \delta \leq 135^o$ then η = $[90^o - (\delta - 90^o)]/90^o$; if $135^o < \delta \leq 180^o$ then η = $(\delta - 90^o)/90^o$

For notation create Figures from examples.

Value

Returns a list with four or five items.

`lines` A matrix with *n* - 1 rows containing line lengths for the lines *A, B, C, D,* and *F*. See figure in examples below.

`angles` A matrix with *n* - 1 rows containing line lengths for the angles κ, γ and δ. See Figure in examples below.

`moment.by.moment` This component provides a matrix with *n* - 1 rows. Included are the columns: `End.time`, `Eta.Index`, `Delta`, `Habitat`, and `Brdr chng`. The columns `Habitat`, and `Brdr chng` are excluded if `habitat = NULL` or `near.angle = NULL`.

`P.summary` Contains averages and standard errors for η.

`crossing.summary` Crossing binomial summaries. Provided if `habitat` and `near.angle` are specified.

Author(s)

Ken Aho

References

Kie, J.G., A.A. Ager, and R.T. Bowyer (2005) Landscape-level movements of North American elk (*Cervus elaphus*): effects of habitat patch structure and topography. *Landscape Ecology* 20:289-300.

McGarigal K., SA Cushman, M.C. Neel, and E. Ene (2002) *FRAGSTATS: Spatial Pattern Analysis Program for Categorical Maps*. Computer software program produced by the authors at the University of Massachusetts, Amherst.

See Also

near.bound, bound.angle

Examples

```
###Diagram describing prp output.
y<-rnorm(100,0,5)
plot(seq(1,100),sort(y),type="l",xaxt="n",yaxt="n",lwd=2,xlab="",ylab="")
par(font=3)

segments(52,-12,46,sort(y)[46],lty=1,col=1,lwd=1)##A
segments(90,-8,85,sort(y)[85],lty=1,col=1,lwd=1)##B
segments(46,sort(y)[46],85,sort(y)[85],lty=1)##F
segments(90,-8,46,sort(y)[46],lty=2)##D
arrows(52,-12,90,-8,length=.1,lwd=3)##C
arrows(20,-12,20,8,lty=2,col="gray",length=.1)#North
arrows(20,sort(y)[46],95,sort(y)[46],length=.1,lty=2,col="gray")
arrows(20,-12,95,-12,length=.1,lty=2,col="gray")#East

text(20,9,"N",col="gray");text(97,-12, "E", col= "gray");text(97,sort(y)[46], "E",
col= "gray")
text(49.5,-12.5,"a");text(92.5,-8.5,"b")
text(45.5,-5.5,"A",font=4,col=1);text(70,-9,"C",font=4,col=1);text(91.5,-1.75,"B",
font=4,col=1)
text(44,sort(y)[46]+1,"c");text(67.5,-2.5,"D",font=4,col=1);text(65,3.9,"F",font=4,
col=1)
text(87,sort(y)[87]+1,"d");text(57,-10,expression(kappa),col=1);
text(81,sort(y)[87]-3,expression(gamma),col=1);text(57,1.3,expression(theta),col=1)
text(64,-11.5,expression(beta),col=1)

library(plotrix)
draw.arc(50,-12,6,1.35,col=1);draw.arc(50,-12,6,.3,col=1);draw.arc(50,-12,6,0.02,
col=1)
draw.arc(46,sort(y)[46],7,.01,col=1);draw.arc(46,sort(y)[46],7,.5,col="white")
draw.arc(85,sort(y)[85],6,-2.7,col=1);draw.arc(85,sort(y)[85],6,-1.4,col="white",
lwd=2)
legend("topleft",c(expression(paste(kappa, " = acos[(",C^2," + ",X^2," - ",D^2,")
/2CX]")),
expression(paste(gamma," = acos[(",Y^2," + ",F^2," - ",D^2,")/2YF]")),
expression(paste(theta," = atan[(",y[f]," - ",y[n],")/(",x[f]," - ",x[n],")]")),
expression(paste(beta, " = atan[(",y[epsilon]," - ",y[alpha],")/(",x[epsilon],
" - ",x[alpha],")]"))),
bty="n",cex=.9,inset=-.025)

###Figure for demo dataset.
bX<-seq(0,49)/46

bY<-c(4.89000,4.88200,4.87400,4.87300,4.88000,4.87900,4.87900,4.90100,4.90800,
4.91000,4.93300,4.94000,4.91100,4.90000,4.91700,4.93000,4.93500,4.93700,
4.93300,4.94500,4.95900,4.95400,4.95100,4.95800,4.95810,4.95811,4.95810,
4.96100,4.96200,4.96300,4.96500,4.96500,4.96600,4.96700,4.96540,4.96400,
```

```
4.97600,4.97900,4.98000,4.98000,4.98100,4.97900,4.98000,4.97800,4.97600,
4.97700,4.97400,4.97300,4.97100,4.97000)

X<-c(0.004166667,0.108333333,0.316666667,0.525000000,0.483333333,0.608333333,
0.662500000,0.683333333,0.900000000,1.070833333)
Y<-c(4.67,4.25,4.26,4.50,4.90,4.10,4.70,4.40,4.20,4.30)

plot(bX,bY,type="l",lwd=2,xlab="",ylab="",ylim=c(4,5.1))
lines(X,Y)

for(i in 1:9)arrows(X[i],Y[i],X[i+1],Y[i+1],length=.1,lwd=1,angle=20)
mx<-rep(1,9)
my<-rep(1,9)
for(i in 1:9)mx[i]<-mean(c(X[i],X[i+1]))
for(i in 1:9)my[i]<-mean(c(Y[i],Y[i+1]))
for(i in 1:9)text(mx[i],my[i],i,font=2,cex=1.3)

nn<-near.bound(X,Y,bX,bY)
prp(seq(1,10),X,Y,nn[,1],nn[,2])$moment.by.moment
```

`pseudo.v` *Jacknife pseudo-values*

Description

The function returns first-order jackknife pseudovalues which can then be used to create statistical summaries, e.g. the jackknife parameter estimate, and the jackknife standard error. The function can be run on univariate data (`matrix = FALSE`) or multivariate data (`matrix =TRUE`). In the later case matrix rows are treated as multivariate observations.

Usage

```
pseudo.v(data, statistic, matrix = FALSE)
```

Arguments

`data`	A vector (`matrix =FALSE`) or matrix (`matrix=TRUE`) of quantitative data.
`statistic`	A function whose output is a statistic (e.g. a sample mean). The function must have only one argument, a call to `data`.
`matrix`	A logical statement. If `matrix = TRUE` then rows in the matrix are sampled as multivariate observations.

Details

In the first order jackknife procedure a statistic $\hat{\theta}$ is calculated using all n samples, it is then calculated with the first observation removed $\hat{\theta}_{-1}$, with only the second observation removed, $\hat{\theta}_{-2}$, and so on. This process is repeated for all n samples. The resulting vector of size n contains pseudovalues for their respective observations.

Value

A vector of first-order jackknife pseudovalues is returned.

Author(s)

Ken Aho

References

Manly, B. F. J. (1997) *Randomization and Monte Carlo methods in biology, 2nd edition*. Chapman and Hall, London.

See Also

empinf, boot, bootstrap

Examples

```
library(vegan)
data(varespec)
# A partial set of observations from a single plot for a Scandinavian
# moss/vascular plant/lichen survey.
site18<-data.frame(t(varespec[1,][1:20]))

#Shannon-Weiner diversity
SW<-function(data){
d<-data[data!=0]
p<-d/sum(d)
-1*sum(p*log(p))
}

pv<-pseudo.v(site18,SW)
```

R.bw *Biweight midvariance, and biweight midcorrelation.*

Description

Calculates biweight midvariance if one variable is given and biweight midvariances, midcovariance and midcorrelation if two variables are given. Biweight midcorrelation is a robust alternative to Pearson's *R*.

Usage

```
R.bw(X, Y=NULL)
```

Arguments

X	A numeric vector
Y	An optional second numeric variable.

Details

Biweight statistics are robust to violations of normality. Like the sample median the sample midvariance has a breakdown point of approximately 0.5. The triefficeincy of the biweight midvariance was the highest for any of the 150 measures of scale compared by Lax (1985).

Value

Returns the biweight variance if one variable is given, and the biweight midvariances, midcovariance and midcorrelation if two variables are given.

Author(s)

Ken Aho

References

Lax, D. A. (1985) Robust estimators of scale: finite sample performance in long-tailed symmetric distributions. *Journal of the American Statistical Association*, 80 736-741.

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

cor, R.pb

Examples

```
x<-rnorm(100)
y<-rnorm(100)
R.bw(x,y)
```

R.pb *Percentage bend correlation*

Description

The percentage bend correlation is a robust alternative to Pearson's product moment correlation.

Usage

```
R.pb(X, Y, beta = 0.2)
```

Arguments

X	A quantitative vector
Y	A second quantitative vector
beta	Bend criterion

Details

The percentage bend correlation belongs to class of correlation measures which protect against marginal distribution (X and Y) outliers. In this way it is similar to Kendall's τ, Spearman's ρ, and biweight midcovariance. A second class of robust correlation measures which take in to consideration the overall structure of the data (O estimators) are discussed by Wilcox (2005, pg. 389). A value for the bend criterion `beta` is required in the `R.pb` function; `beta` = 0.2 is recommended by Wilcox (2005).

Value

A dataframe with the correlation, test statistic and p-value for the null hypothesis of independence are returned.

Author(s)

Ken Aho

References

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`corr`, `R.bw`

Examples

```
x<-rnorm(100)
y<-rnorm(100)
R.pb(x,y)
```

`radiation.heatl` *Radiation-heatload algorithm*

Description

The function radiation.heatl calculates annual incident solar radiation (MJ cm^2 yr$^-1$) and heatload (a radiation index based on the idea that highest amounts of radiation occur on southwest facing slopes in the northern hemisphere).

Usage

```
radiation.heatl(slope, aspect, lat)
```

Arguments

`slope`	slope (measured in degrees)
`aspect`	aspect (measured in degrees)
`lat`	latitude (measured in degrees)

Details

The function requires three types of data: `slope` (measured in degrees), `aspect` (measured in degrees), and northern latitude = `lat` (measured in degree). The function is based on equations from a paper written by McCune and Keon (2002). Note that this function ignores climatic factors including cloudiness, and is therefore is probably best for relative comparisons of radiation and heatload within a region and not for absolute measurements.

Value

`radiation`	Annual incident radiation in (MJ cm^2 yr^-1)
`heatload`	A unitless measure of heatload, maximized on SW aspects.

Author(s)

Ken Aho

References

McCune, B., and D. Keon (2002) Equations for potential annual direct radiation and heat load. *Journal of Vegetation Science*, 13: 603-606.

Examples

```
slope<-20
asp<-25
lat<-46
radiation.heatl(slope,asp,lat)
```

`rankindex.new` *Compares the efficacy of resemblance measures for a particular data scenario.*

Description

This function is a wrapper for `rankindex` from package **vegan**.

Usage

```
rankindex.new(grad, veg, indices = c("matching", "rogers", "jaccard.pa",
"sorenson", "kulkczynski.pa", "ochiai", "gower", "steinhaus", "kulkczynski.q",
"jaccard.q", "euclidean", "rel.euclidean", "manhattan", "czekanowski",
"whittaker", "canberra", "chi.metric", "chi.dist", "morisita", "morisita.horn",
"minkowski", "mountford", "raup.crick", "binomial", "chao"), stepacross = FALSE,
 method = "pearson", ...)
```

Arguments

`grad`	An *s* x *e* matrix of environmental variables, where *s* = the number of sites and *e* = the number of environmental variables.
`veg`	An *s* x *p* community matrix, where *s* = the number of sites and *p* = the number of species.
`indices`	Some subset from the character string: `c("matching", "rogers", "jaccard.pa", "sorenson", "kulkczynski.pa", "ochiai", "gower", "steinhaus", "kulkczynski.q", "jaccard.q", "euclidean", "rel.euclidean", "manhattan", "czekanowski", "whittaker", "canberra", "chi.metric", "chi.dist", "morisita", "morisita.horn", "minkowski", "mountford", "raup.crick", "binomial", "chao")`.
`stepacross`	Logical, specifies whether a step across transformation should be applied to the resemblance matrix.
`method`	What method is used to calculate correlations between environmental and communtiy matrices. Must be one of `"pearson"`, `"kendall"`, or `"spearman"`.
`...`	Other parameters to `stepacross` from vegan.

Details

A number of ecologists have evaluated the performance of distance measures by comparing environmental distance (i.e. differences along gradients) to distance in ordination space (Beals, 1984; Faith et al., 1987; De'ath, 1999). If species distributions are well described by environmental variables, then a strong association should exist between distances in species space and differences in environmental space. Thus, poor correlations represent poor performance by the distance measure. Obviously this analysis becomes more meaningful as the number of important environmental variables increases in ones measure of environmental distance. The library vegan has a function called rankindex which ranks dissimilarity or distances used for finding community distances or dissimilarities by how well these indices agree with gradient differences. The gradient separation between each point is expressed as Euclidean distance for continuous variables and as Gower's metric for mixed data (i.e. when at least some environmental variables are categorical or ordinal). In the later case the library **cluster** is required. The association of community and environmental distance matrices is simply the correlation of the community and environmental distance ranks and can be measured with any of the conventional measures described in Ch. 11. The function `rankindex.new` is a wrapper for `rankindex` and uses Oksanen's method to compare the efficacy of 25 of the 26 indices generated by `get.dist`. Mahalanobis distance is left out, since it does not create a distance matrix *per se*, but a simultaneous comparison of each site to all other sites.

Value

The function returns a table of ranked Pearson's correlations (default) and a barplot.

Author(s)

Ken Aho

References

Oksanen, J., Kindt, R., Legendre, P., O'Hara, B., Simpson, G. L., and Stevens, M. H. H. (2008) *vegan: community ecology package. R package version 1.13-0*. http://vegan.r-forge.r-project.org/

See Also

get.dist

Examples

```
## Not run:
library(vegan)
data(varechem)
data(varespec)
r<-rankindex.new(scale(varechem),varespec)
## End(Not run)
```

`rat` *Rat glycogen data from Sokal and Rohlf (1995).*

Description

This dataset from Sokal and Rohlf (1995) is often used to demonstrate psuedoreplication. Six rats were randomly given one of three treatments: "control", "compound 217", and "compound 217 + sugar". After a short period of time the rats were euthanized and the glycogen content of their livers was measured. Two glycogen measurements were made for three different preparations of each liver. Clearly the liver preparations and measurements on those preparations are nested in each rat, and are not independent.

Usage

```
data(rat)
```

Format

A data frame with 36 observations on the following 4 variables.

`Glycogen` A numeric vector describing glycogen levels. Units are arbitrary.

`Treatment` A character vector describing the type of nutritional compound being used 1 = "control", 2 = "compound 217", 3 = "compound 217 + sugar".

`Rat` Rat number assigned to each treatment

`Liver` Portion of liver used in analysis

Source

Sokal, R. R., and Rohlf, F. J. (1995) *Biometry*. W. H. Freeman and Co., New York.

`refinery` *Refinery CO dataset*

Description

In the early 1990s an oil refinery northeast of San Francisco agreed with local air quality regulators [the Bay Area Air Quality Management District (BAAQMD)] to reduce carbon monoxide emissions. Baselines for reductions were to be based on measurements of CO made by refinery personnel, and by independent measurements from BAAQMD scientists for the roughly the same time period

Usage

```
data(refinery)
```

Format

The dataframe contains three columns:

CO Carbon monoxide. Measured in ppm.

Source The source of measurements; either refinery or BAAQMD.

Date Month/Day/Year

Source

http://lib.stat.cmu.edu/DASL/Stories/MeasuringAirPollution.html; accessed 6/29/110 samp.dist *Animated and/or snapshot representations of a statistic's sampling distribution*

Description

This help page describes a series of **asbio** functions for depicting sampling distributions. The function `samp.dist` samples from a parent distribution without replacement with sample size = `s.size`, `R` times. At each iteration a statistic requested in `stat` is calculated. Thus a distribution of `R` statistic estimates is created. This distribution is shown as an animated or non-animated relative frequency histogram. Sampling distributions for up to four different statistics utilizing two different parent distributions are possible. Sampling distributions can be combined in various ways by specifying a function in `func` (see below). The function `samp.dist.samp` creates snapshots, i.e. simultaneous views of a sampling distribution at particular sample sizes.

Usage

```
samp.dist(parent, parent2 = parent, s.size = 1, s.size2 = s.size,
n.seq = seq(1, 30),R = 1000, breaks = 30, stat = mean, stat2 = NULL,
stat3 = NULL, stat4 = NULL, fix.n = TRUE, xlab = expression(bar(x)),
ylab = "Relative frequency", ylim = NULL, func = NULL, show.n = TRUE,
show.SE = FALSE, est.density = TRUE, col.density = 4, lwd.density = 2,
est.ylim = TRUE, anim = TRUE, interval = 0.01, col.anim = "rainbow",
digits = 3,...)

samp.dist.fixn(parent, parent2 = parent, s.size = 1, s.size2 = s.size,
R = 1000, breaks = 30, stat = mean, stat2 = NULL, stat3 = NULL,
stat4 = NULL, xlab = expression(bar(x)), ylab = "Relative frequency",
func = NULL, show.n = TRUE, show.SE = FALSE, anim = TRUE,
interval = 0.01, col.anim = "rainbow", digits = 3,...)

samp.dist.n(parent, R = 500, n.seq = seq(1, 30), stat = mean,
xlab = expression(bar(x)), ylab = "Relative frequency", breaks = 30,
func = NULL, show.n = TRUE, show.SE = FALSE, est.density = TRUE,
col.density = 4, lwd.density = 2, est.ylim = TRUE, ylim = NULL,
anim = TRUE, interval = 0.5, col.anim = NULL, digits = 3,...)

samp.dist.tck()

samp.dist.method.tck()

samp.dist.snap(parent,parent2 = NULL,stat = mean,stat2 = NULL, stat3 =
NULL, stat4 = NULL,s.sizes=c(1,3,6,10,20,50),s.sizes2 = NULL,R = 1000,L = 6,
xlab = expression(bar(x)),show.SE = TRUE)

samp.dist.snap.tck()
```

Arguments

parent	A vector containing observations from a parental distribution. For computational efficiency datasets exceeding 100000 observations are not recommended.
parent2	An optional second parental distribution, useful for the construction of sampling distributions of test statistics.
s.size	Sample size to be taken at each of R iterations from the parental distribution.
s.size2	An optional second sample size if a second statistic is to be calculated.
s.sizes	An argument from samp.dist.snap. This will be a vector of maximum length twelve indicating which sample sizes (taken from parent) should be simultaneously depicted.
s.sizes2	An optional argument from samp.dist.snap. This will be a vector of maximum length twelve indicating which sample sizes (taken from parent2) should be simultaneously depicted. Must be the same length as s.sizes.
n.seq	A two element vector specifying the smallest and largest in a range of demonstrated sample sizes. Requires fix.n=FALSE.
L	An argument from samp.dist.snap. Will be a single integer indicating the length of s.sizes (i.e. the number of desired snapshots), and ,if specified, the length of s.sizes2.
R	The number of samples to be taken from parent distribution(s).
breaks	Number of breaks in the histogram.
stat	The statistic whose sampling distribution is to be represented. Will work for any summary statistic; e.g. mean, var, median, etc.
stat2	An optional second statistic. Useful for conceptualizing sampling distributions of test statistics. Calculated from sampling parent2.
stat3	An optional third statistic. The sampling distribution is created from the same sample data used for stat.
stat4	An optional fourth statistic. The sampling distribution is created from the same sample data used for stat2.
fix.n	Logical indicating whether or not sample size should be held constant in demonstrations (see below).
xlab	X-axis label.
ylab	Y-axis label.
ylim	Limits for Y-axis. Specify using a two element vector.
func	An optional function used to manipulate a sampling distribution or to combine the sampling distributions of two or more statistics. The function must have only sampling distributions, i.e. s.dist1, s.dist2, s.dist3, and/or s.dist as non-fixed arguments (see example below).
show.n	A logical command, TRUE indicates that sample size for parent will be displayed.
show.SE	A logical command, TRUE indicates that bootstrap standard error for the statistic will be displayed.

est.density	A logical command, if TRUE then a density line is plotted over the histogram. Only used if fix.n = true.
col.density	The color of the density line. See est.density above.
lwd.density	The width of the density line. See est.density above.
est.ylim	Logical. If TRUE Y-axis limits are estimated logically for the animation. Consistent Y-axis limits make animations easier to visualize. Only used if fix.n = TRUE.
anim	A logical command indicating whether or not animation should be used.
interval	Animation speed. Decreasing interval increases speed.
col.anim	Color to be used in animation. Three changing color palettes: rainbow, gray, heat.colors, or "fixed" color types can be used.
digits	The number of digits to be displayed in the bootstrap standard error.
...	Additional arguments from plot.histogram.

Details

Sampling distributions of individual statistics can be created, or the function can be used in more sophisticated ways, e.g. to create sampling distributions of ratios of statistics, i.e. t^*, F^* etc. (see examples below). To provide pedagogical clarity animation for figures is provided.

Three general uses of the function samp.dist are possible.

1) One can demonstrate the accumulation of statistics for a single sample size using animation. This is useful because as more and more statistics are acquired the frequentist paradigm associated with sampling distributions becomes better represented (i.e the number of estimates is closer to infinity). This is elucidated by allowing the default fix.n = TRUE. Animation will be provided with the default anim = TRUE. This approach also allows specification of up two parent distributions, up to two sample sizes, and up to four distinct statistics (i.e. four distinct sampling distributions, representing four distinct estimators can be created). The arguments stat and stat3 will be drawn from parent, while stat3 and stat4 will be drawn from parent2. These distributions can be manipulated and combined in an infinite number of ways with an auxiliary function called in the argument func (see examples below). This allows depiction of sampling distributions made up of multiple estimators, e.g. test statistics.

2) One can provide a seamless animated demonstration of the effect of varying sample size on a sampling distribution by specifying n.fixed = FALSE. If n.fixed = FALSE is used, then a range of sample sizes (integers) must also be specified as a two element vector in n.seq. Note if n.fixed = FALSE then the arguments s.size and s.size2 will be superfluous. In addition, multiple statistics and parent populations are not supported if n.fixed = FALSE, although auxilary functions can still be called with func.

3) One can provide simultaneous snapshots of a sampling distribution at a particular sample size with the function samp.dist.snap.

Loading the package **tcltk** allows use of the functions samp.dist.tck, samp.dist.method.tck, and samp.dist.snap.tck, which provide interactive GUIs that run samp.dist.

Value

Returns a representation of a statistic's sampling distribution in the form of a histogram.

Author(s)

Ken Aho

See Also

plot.histogram, hist, bootstrap.

Examples

```
## Not run:
##Central limit theorem
#Snapshots of four sample sizes.
exp.parent<-rexp(100000)
samp.dist.snap(parent=exp.parent, s.sizes=c(1,5,10,50), R=1000)

##n not fixed -- sample mean
exp.parent<-rexp(10000)
samp.dist(parent=exp.parent, col.anim="heat.colors",fix.n=FALSE,interval=.3)

##n not fixed -- sample mean and sample median (both are consistent and unbiased,
# but which is more efficient for mu?).
# This will take a few seconds.
parent<-rnorm(10000,sd=3)
dev.new()
samp.dist(parent, R=1000,col.anim="heat.colors",fix.n=FALSE,interval=.1,
n.seq=seq(1,100),breaks=50,show.SE=TRUE)
dev.new()
samp.dist(parent, R=1000,col.anim="heat.colors",fix.n=FALSE,interval=.1,stat=median,
n.seq=seq(1,100),xlab="Median",show.SE=TRUE,breaks=50)

#How do the efficiency of the median and mean compare in a distribution with 10%
#contamination?
parent<-c(rnorm(9000),rnorm(1000,mean=10))
samp.dist(parent, col.anim="heat.colors",fix.n=FALSE,interval=.3,
breaks=50,show.SE=TRUE)
dev.new()
samp.dist(parent, col.anim="heat.colors",fix.n=FALSE,interval=.3,stat=median,
xlab="Median",est.ylim=TRUE,show.SE=TRUE,breaks=50)


##Distribution of t-statistics from a pooled variance t-test under valid and invalid assumptions
#valid
parent<-rnorm(100000)
t.star<-function(s.dist1,s.dist2,s.dist3,s.dist4,s.size=6,s.size2=s.size){
MSE<-(((s.size-1)*s.dist3)+((s.size2-1)*s.dist4))/(s.size+s.size2-2)
func.res<-(s.dist1-s.dist2)/(sqrt(MSE)*sqrt((1/s.size)+(1/s.size2)))
func.res}

samp.dist(parent, parent2=parent, s.size=6, R=1000, breaks=35,stat=mean,stat2=mean,
stat3=var,stat4=var,xlab="t*", ylab="Relative frequency",func=t.star,show.n=FALSE)

curve(dt(x,10),from=-6,to=6,add=TRUE,lwd=2)
```

```
legend("topleft",lwd=2,col=1,legend="t(10)")

#invalid; same population means (null true) but different variances and other distributional
#characteristics.
parent<-runif(100000, min=0,max=2)
parent2<-rexp(100000)

samp.dist(parent, parent2=parent2, s.size=6, R=1000, breaks=35,stat=mean,stat2=mean,
stat3=var,stat4=var,xlab="t*", ylab="Relative frequency",func=t.star,show.n=FALSE)
curve(dt(x,10),from=-6,to=6,add=TRUE,lwd=2)
legend("topleft",lwd=2,col=1,legend="t(10)")

#Interactive GUI, require package 'tcltk'
samp.dist.tck()
samp.dist.snap.tck()

## End(Not run)
```

---

sc.twin *Matched pairs schizophrenia data*

Description

Scientists have long been concerned with identifying physiological characteristics which result in a disposition for schizophrenia. Early studies suggested that the volume of particular brain regions of schizophrenic patients may differ from non-afflicted individuals. However these studies often contained confounding variables (e.g. socioeconomic status, genetics) which obscured brain volume/ schizophrenia relationships (Ramsey and Schafer 1997). To control for confounding variables Suddath et al. (1990) examined 15 pairs of monozygotic twins where one twin was schizophrenic and the other was not. Twins were located from an intensive search throughout the United States and Canada. The authors used magnetic resonance imaging to measure brain volume of particular regions in the twin's brains.

Usage

```
data(sc.twin)
```

Format

The dataframe has 2 columns:

unaffected Left hippocampus volumes for unaffected twins.

affected Left hippocampus volumes for affected twins.

`see.exppower.tck` *Visualize exponential power functions*

Description

Visualize exponential power functions, including a Gaussian distribution.

Usage

```
see.exppower.tck()
```

Details

The normal distribution and Gaussian distribution are based on an exponential power function:

$$f(x) = exp(-|x|^m)$$

Letting $m = 2$ results in a Gaussian distribution. Standardizing this so that the area under the curve = 1 results in the standard normal distribution.

Author(s)

Ken Aho

See Also

`demos`

`see.HW` *Visualize the Hardy Weinberg equlibrium*

Description

Allows interactive depiction of the Hardy Weinberg equilibrium.

Usage

```
see.HW(parg)
see.HW.tck()
```

Arguments

`parg`	Proportion of the allele *p* in the population, i.e. a number between 0 and 1.

Details

Solves and depicts the Hardy Weinberg equilibrium, i.e:

$$pp + 2pq + qq = 1$$

Author(s)

Ken Aho

`see.logic` *Interactive worksheet for logical and fallacious arguments*

Description

It is vital that scientists understand what logical and fallacious arguments are. This worksheet provides a predagogical tool for logic.

Usage

```
see.logic()
```

Author(s)

Ken Aho

References

Salmon, W (1963) *Logic*. Prentice-Hall

See Also

demos

Examples

```
## Not run:
see.logic()

## End(Not run)
```

see.M — *Visualization of the M-estimation function*

Description

The function provides interactive visualization of robust *M* estimation of location.

Usage

```
see.M()
```

Details

The value c = 1.28 gives 95 percent efficiency of the mean given normality. The sample median and mean can be considered special cases of *M*-estimators. The value c = 0 provides the sample median, while the value, $c = \infty$ gives the sample mean.

Author(s)

Ken Aho

References

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

huber.mu, huber.NR

see.nlm — *Visualize important non-linear functions*

Description

A number of important equation forms require that their parameters be estimated using the non-linear least squares. Here are six.

Usage

```
see.nlm()
```

Author(s)

Ken Aho

References

Crawley, M. J. (2007) *The R book*. Wiley.

`see.norm.tck` *Visualize pdfs*

Description

Interactive GUIs for visualizing how distributions change with changing values of pdf parameters, e.g. μ and σ. The basic ideas here are lifted largely from a clever function from Greg Snow's package **TeachingDemos**. The functions `see.pdfdriver.tck` and `see.pdfdriver` are **tcltk** utility functions.

Usage

```
see.norm.tck()
see.normcdf.tck()
see.beta.tck()
see.betacdf.tck()
see.bin.tck()
see.bincdf.tck()
see.chi.tck()
see.chicdf.tck()
see.exp.tck()
see.expcdf.tck()
see.F.tck()
see.Fcdf.tck()
see.gam.tck()
see.gamcdf.tck()
see.geo.tck()
see.geocdf.tck()
see.hyper.tck()
see.hypercdf.tck()
see.logis.tck()
see.logiscdf.tck()
see.nbin.tck()
see.nbincdf.tck()
see.lnorm.tck()
see.lnormcdf.tck()
see.pois.tck()
see.poiscdf.tck()
see.t.tck()
see.tcdf.tck()
see.unif.tck()
see.unifcdf.tck()
see.weib.tck()
```

```
see.weibcdf.tck()
see.pdfdriver.tck()
see.pdfdriver(pdf, show.cdf = TRUE)
```

Arguments

`pdf`	Name of probability density function
`show.cdf`	Logical, indicating whether or not the cumulative distribution function should be shown.

Author(s)

Ken Aho

Examples

```
## Not run:
see.norm.tck()

## End(Not run)
```

`see.power` *Interactive depiction of type I and type II error and power*

Description

Provides an interactive pedagogical display of power.

Usage

```
see.power(alpha = NULL, sigma = NULL, n = NULL, effect = NULL,
test = "lower", xlim = c(-3, 3), strict = FALSE)

see.power.tck()
```

Arguments

`alpha`	Type I error.
`sigma`	Standard deiviation of underlying population.
`n`	sample size
`effect`	Effect size
`test`	Type of test, one of `c("lower","upper","two")`.
`xlim`	*X*-axis limits
`strict`	Causes the function to use a strict interpretation of power in a two-sided test. If strict = TRUE then power for a two sided test will include the probability of rejection in the opposite tail of the true effect. If strict = FALSE (the default) power will be half the value of α if the true effect size is zero.

Details

The function `see.power` provides an interactive display of power. The function `see.power.tck` provides a **tcltk** GUI to manipulate `see.power`

Author(s)

Ken Aho

`see.regression` *Interactive visualization of least squares regression.*

Description

Scatterplot points can be moved with `see.regression`, while points can be added and deleted with `see.adddel`. The function `see.regression` is an appropriation from **tcltk** demos, with a few bells and whistles added.

Usage

```
see.regression()
see.adddel()
```

Author(s)

the R Development Core Team for `see.regression`, Ken Aho for `see.adddel`.

`see.typeI_II` *Interactive depiction of type I and II error*

Description

The function provides a **tcltk** driver illustrating type I, type II error, and power.

Usage

```
see.typeI_II()
```

Author(s)

Ken Aho

See Also

`see.power`,`power.z.test`

`selftest.se.tck1` *Interactive self-testing statistical questions*

Description

These functions provide interactive multiple-choice questions that can be used to emphasize pedagogical ideas.

Usage

```
selftest.se.tck1()
selftest.se.tck2()
selftest.conf.tck1()
selftest.conf.tck2()
selftest.pdfs.tck1()
selftest.pdfs.tck2()
selftest.prob.tck1()
selftest.prob.tck2()
selftest.prob.tck3()
selftest.sampd.tck1()
selftest.sampd.tck2()
```

Author(s)

Ken Aho

`Semiconductor` *Split plot computer chip data from Littell et al. (2006).*

Description

Littell et al. (2006) use the data here to introduce analysis of split plot designs using mixed models. Twelve silicon wafers were randomly selected from a lot, and were randomly assigned to four different processing modes. Resistance on the chips was measured in four different positions (four different chips) on each wafer. Mode of processing and position of chips were fixed factors, while wafer was a random effect. The experimental units with respect to process are the wafers. The experimental units with respect to position are individual chips. Thus the wafer is the whole plot, while the positions (chips) are split plot units

Usage

```
data(Semiconductor)
```

Format

The dataframe contains four columns:

Resistance The response variable of interest. Measured in ohms.

Process The explanatory variable of interest. The type of process used to create the computer chips. A factor with 4 levels.

Wafer The whole plot containing four chips. There were four wafers tested, i.e. four levels, `1,2,3,4`.

Chip Position on the wafer. These are split plots within the whole plots. Four levels: `1,2,3,4`.

Source

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and O. Schabenberger. 2006. *SAS for mixed models 2nd ed.* SAS press.

`shade.norm` *Shading functions for interpretation of pdf probabilities.*

Description

Creates plots with lower, upper, two-tailed, and middle of the distribution shading for popular pdfs.

Usage

```
shade.norm(x = NULL, from = NULL, to = NULL, sigma = 1, mu = 0,
tail = "lower", show.p = TRUE, show.d = FALSE, show.dist = TRUE, digits = 5,
shade.col="gray",...)

shade.t(x = NULL, from = NULL, to = NULL, nu = 3, tail = "lower",
show.p = TRUE, show.d = FALSE, show.dist = TRUE, digits = 5,
shade.col="gray",...)

shade.F(x = NULL, from = NULL, to = NULL, nu1 = 1, nu2 = 5,
tail = "lower", show.p = TRUE, show.d = FALSE, show.dist = TRUE,
prob.to.each.tail = 0.025, digits = 5, shade.col="gray",...)

shade.chi(x = NULL, from = NULL, to = NULL, nu = 1, tail = "lower",
show.p = TRUE, show.d = FALSE, show.dist = TRUE, prob.to.each.tail = 0.025,
digits = 5,shade.col="gray",...)

shade.bin(x=NULL,from=NULL,to=NULL,n=1,p=0.5,tail="X=x",show.p=TRUE,
show.dist=TRUE, show.d=FALSE,digits=5, ...)

shade.poi(x=NULL,from=NULL,to=NULL,lambda=5,tail="X=x",show.p=TRUE,
show.dist=TRUE, show.d=FALSE,digits=5, ...)
```

Arguments

x	A quantile, i.e. $X = x$.
from	To be used with `tail = "middle"`; the value *a* in $P(a < X < b)$.
to	To be used with `tail = "middle"`; the value *b* in $P(a < X < b)$.
sigma	Standard deviation for the nomral distribution.
mu	Mean of the normal distribution.
tail	One of four possibilities: `"lower"` provides lower tail shading, `"upper"` provides upper tail shading, `"two"` provides two tail shading, and `"middle"` provide shading in the middle of the pdf, between `"from"` and `"to"`. For discrete pdfs (binomial and Poisson) the possibility "X=x" is also allowed, and will be equivalent to the density. Two tailed probability is not implemented for `shade.poi`.
show.p	Logical; indicating whether probabilities are to be shown.
show.d	Logical; indicating whether densities are to be shown.
show.dist	Logical; indicating whether parameters for the distribution are to be shown.
nu	Degrees of freedom.
nu1	Numerator degrees of freedom for the *F*-distribution.
nu2	Denominator degrees of freedom for the *F*-distribution.
prob.to.each.tail	Probability to be apportioned to each tail in the *F* and Chi-square distributions if `tail = "two"`.
digits	Number of digits to be reported in probsabilities and densities.
n	The number of trials for the binomial pdf.
p	The binomial probability of success.
lambda	The Poisson parameter (i.e. rate).
shade.col	Color of probability shading.
...	Additional arguments to `plot`.

Value

Returns a plot with the requested pdf and probability shading.

Note

Lower-tailed chi-squared probabilities are not plotted correctly for df < 3.

Author(s)

Ken Aho

Examples

```
## Not run:
##normal
shade.norm(x=1.2,sigma=1,mu=0,tail="lower")
shade.norm(x=1.2,sigma=1,mu=0,tail="upper")
shade.norm(x=1.2,sigma=1,mu=0,tail="two")
shade.norm(from=-.4,to=0,sigma=1,mu=0,tail="middle")
shade.norm(from=0,to=0,sigma=1,mu=0,tail="middle")

##t
shade.t(x=-1,nu=5,tail="lower")
shade.t(x=-1,nu=5,tail="upper")
shade.t(x=-1,nu=5,tail="two")
shade.t(from=.5,to=.7,nu=5,tail="middle")

##F
shade.F(x=2,nu1=15,nu2=8,tail="lower")
shade.F(x=2,nu1=15,nu2=8,tail="upper")
shade.F(nu1=15,nu2=8,tail="two",prob.to.each.tail=0.025)
shade.F(from=.5,to=.7,nu1=15,nu2=10,tail="middle")

##Chi.sq
shade.chi(x=2,nu=5,tail="lower")
shade.chi(x=2,nu=5,tail="upper")
shade.chi(nu=7,tail="two",prob.to.each.tail=0.025)
shade.chi(from=.5,to=.7,nu=5,tail="middle")

##binomial
shade.bin(x=5,n=20,tail="X=x",show.d=TRUE)
shade.bin(x=5,n=20,tail="lower")
shade.bin(x=5,n=20,tail="two")
shade.bin(from=8,to=12,n=20,tail="middle")

##Poisson
shade.poi(x=5,lambda=6,tail="X=x",show.d=TRUE)
shade.poi(x=5,lambda=7,tail="lower")
shade.poi(x=5,lambda=8,tail="upper")
shade.poi(from=8,to=12,lambda=7,tail="middle")

## End(Not run)
```

`shade.norm.tck` *GUI display of probability*

Description

Provides **tcltk** GUIs to manage **asbio** shade functions.

Usage

```
shade.bin.tck()

shade.chi.tck()

shade.F.tck()

shade.norm.tck()

shade.poi.tck()

shade.t.tck()
```

Author(s)

Ken Aho

See Also

`shade`

`Shep.comp` *Shepard plot type comparisons for PCoA*

Description

Calculates raw R^2, Linear fit R^2 and non-metric R^2 for PCoA ordinations.

Usage

```
Shep.comp(dis, max.dim = 10)
```

Arguments

`dis`	A dissimilarity matrix. Any index from `get.dist` can be used.
`max.dim`	The maximum number of `cmdscale` dimensions for which correlations should be calculated.

Details

The function Shep.comp in asbio creates calculates three sorts correlations coefficients across all specified PCoA dimensionalities.

1) Pearson's correlations of observed distances (in the distance matrix) and fitted distances (in a PCoA ordination); i.e. "raw fits."

2) Pearson's correlations between the monotonic fitted line and the observed distances "linear fits".

3) A correlation based on stress and calculated as 1 - S^2 (Oksanen 2008), where:

$$S = \sqrt{\frac{\sum_{i \neq j} \hat{d}_{ij} - d_{ij}^2}{\sum_{i \neq j} {d_{ij}}^2,}}$$

where $\hat{d}_{ij}$ is the distance between rank order of distance between point *i* and *j* in the final configuration (i.e. the fitted monotonic stressplot line), and d_{ij} is the distance between point *i* and *j* in the original distance matrix.

The first type of correlation "raw fits" should probably not be used since the relationship between community dissimilarity and a configuration may be strong while being non-linear. The second correlation should be linear even if the relationship between observed dissimilarity and dissimilarities in the final configuration are non-linear, and is often referred to as the linear fit (Oksanen 2008). The final type of correlation has been called "non-metric" fit (Oksanen 2008).

Value

Returns a matrix with three columns containing R^2 for raw, linear, and non-metric R^2's.

Author(s)

Ken Aho

References

Oksanen, J. (2008) *Multivariate analysis of ecological communities in R: vegan tutorial*. http://cc.oulu.fi/~jarioksa/opetus/metodi/vegantutor.pdf

See Also

`get.dist`, `cmdscale`

Examples

```
data(varespec)
d<-get.dist(varespec,"steinhaus")
Shep.comp(d,max.dim=10)
```

---

skew *Sample skewness and kurtosis*

Description

Functions for skewness and kurtosis.

Usage

```
skew(x,method="unbiased")

kurt(x,method="unbiased")
```

Arguments

`x`	A vector of quantitative data.
`method`	The type of method used for computation of skew and kurtosis. Two choices are possible for skewness: `"moments"` and `"unbiased"`, and three choices are possible for kurtosis: `"unbiased", "moments"`, and `"excess"`.

Details

Aside from centrality and variability we can describe distributions with respect to their shape. Two important shape descriptors are skewness and kurtosis. Skewness describes the relative density in the tails of a distribution while kurtosis describes the peakedness of a distribution. When quantified for a population skewness and kurtosis are denoted as γ_1 and γ_2 respectively. For a symmetric distribution skewness will equal zero; i.e. γ_1 = 0. A distribution with more density in its right-hand tail will have γ_1 > 0, while one with more density in its left-hand tail will have γ_1 < 0. These distributions are often referred to as positively-skewed and negatively-skewed respectively. If a distribution is normally peaked (mesokurtic) then γ_2 = 3. As a result the number three is generally subtracted from kurtosis estimates so that a normal distribution will have γ_2 = 0 . Thus strongly peaked (leptokurtotic) distributions will have γ_2 > 0, while flat-looking (platykurtotic) distributions will have a kurtosis γ_2 < 0.

Several types of skewness and kurtosis estimation are possible.

For method of moments estimation let:

$$m_i = (1/n) \sum_i X_i - \bar{X}^i,$$

then the method of moments skewness is: $m_3/m_2^{3/2}$, the method of moments kurtosis is: m_4/m_2^2, and the excess method of moments kurtosis is $m_4/m_2^2 - 3$.

These estimators are biased low, particularly given small sample sizes. A more complex estimator is required to account for this bias. This is provided by `method = "unbiased"` in `skew` and `kurt`.

Value

Output will be the sample skewness or kurtosis.

Author(s)

Ken Aho

Examples

```
exp<-rexp(10000)
skew(exp)
kurt(exp)
```

---

so2.us.cities — *SO2 data for 32 US cities with respect to 6 explanatory variables*

Description

Of concern for public health officials and biologists are models of air pollution as a function of environmental characteristics. Using a meta-analysis of government publications Sokal and Rolf (1995) compiled an interesting dataset which investigates air pollution (measured as annual mean SO_2 concentration per m^3) as a function of six environmental variables for 32 cities in the United States. Whenever the data were available they are based on averages of three years 1969, 1970, and 1971.

Usage

```
data(so2.us.cities)
```

Format

The dataset contains 8 variables:

City US city.

Y Average annual SO_2 concentration per m^3.

X1 Average annual temperature (degrees Celsius).

X2 Number of industrial companies with more than 20 employees.

X3 Population size (1970 census) in thousands.

X4 Average Annual average wind speed.

X5 Average Annual precipitation (cm).

X6 Average number of days with precipitation.

Source

Zar, J. H. (1999) *Biostatistical Analysis, 4th ed.* Prentice-Hall.

`sortid` *Sorts releve table rows using a dot product approach.*

Description

Sorts species(rows) in a releve or summarized releve table assuming a relevant order of columns. A dot product approach described below is used.

Usage

```
sortid(Y, gradient = 1)
```

Arguments

`Y`	An site x species (n x p) community matrix.
`gradient`	A constant used to establish the dot product see discussion below.

Details

A large number of ecologists have used sorted tables to describe patterns of data across gradients (Braun-Blanquet 1964, Hill 1973) . Perhaps the most common method to order a data matrix is to: 1) arrange columns (sites) in an order representing positions along a gradient (e.g. dry to wet), then 2) sort the species according to their dominance within the arranged columns. Hand sorting tables becomes burdensome for large complex datasets, and a large degree of subjectivity will be introduced (i.e. where does one put multimodal, randomly, or uniformly distributed species?). As a result, a number of methods have been developed to automatically sort the order of rows (species) with respect to the gradient represented by the order of columns (e.g. Tichy 2002). One method is to use weighted averaging so that species abundances are weighted by responses of the environmental variable determining the order of columns (Ter Braak and Looman 1986). While this method often produces good results, it is hampered by the distribution shape of the gradient (Jongman 1995). To address this problem I propose using a dot product algorithm (Stewart 2003, pg. 807) to find the order of rows. Consider an n x s matrix **M**, where s = the number of species, n = the number of sites, and c is a constant.

1. Create a vector, **v**, of length n with uniformly spaced intervals from -c to +c. Thus, if$n$ = 5, and c = 1, then **v** = (-1.0, -0.5, 0.0, 0.5, 1.0).

2. Take the dot product of **v** and the vector of abundances of spp 1 from sites 1 through n.

3. Create the vector, **d**, by calculating its elements $d_1, d_2, ..., d_s$, using step 2 above.

4. Sort the rows in **M** with the respect to descending values of **d**.

Note that **v** need not be uniformly distributed (see step 1 in the algorithm description above). To account for distribution shape in the gradient, any distribution (e.g. normal, lognormal) may be used.

Value

A sorted table is returned.

Author(s)

Ken Aho

References

Braun-Blanquet, J. (1964) *Pflanzenoziologia. Grunduge der Vegetationskunde. 3*. Aufl. Berlin. Wien. Springer Verlag, New York, USA.

Hill, M. O. (1979) *TWINSPAN-A FORTRAN program for arranging multivariate data in an ordered two-way table by classification of the individuals and attributes*. Ecology and Systematics. Cornell University, Ithaca, NY, US.

Jongman, R. H. G., ter Braak, C. F. G., and O. F. R. van Tongern. (1995) *Data analysis In Community and landscape ecology*. Cambridge University Press. Cambridge, UK.

Stewart, J. (2003) *Calculus: early transcendentals*. Thompson Learning Inc.

ter Braak, C. J. F., and C. W. N. Looman. (1986) Weighted averaging, logistic regression and the Gaussian response model. *Vegetatio*. 65: 3-11.

Tichy, L. (2002) JUICE, software for vegetation classification. *Journal of Vegetation Science*. 13: 451-453.

See Also

veg.table

Examples

```
data(dune)
sortid(t(dune))
```

---

stan.error — *Variance and standard error estimators for the sampling distribution of the sample mean*

Description

Estimator for the variance for the sampling distribution of the sample mean, i.e. S^2/n, and the standard deviation of the sampling distribution, i.e. $S/\sqrt(n)$.

Usage

```
stan.error(x)
stan.error.sq(x)
```

Arguments

x	A vector of quantitative data describing a sampling distribution.

Author(s)

Ken Aho

See Also

sd

starkey	*DEM data from the Starkey experimental forest in NE Oregon.*

Description

UTM northing and easting data along with 18 other environmental variables describing the Starkey experimental forest.

Usage

```
data(starkey)
```

Source

www.fs.fed.us/pnw/starkey/data/tables/Starkey_OR_Main_Habitat_1993-1996_Data.txt

transM	*Transition matrix analysis*

Description

Creates a plot showing expected numbers of individuals in specified age classes or life stages given survivorship probabilities from a transition matrix (cf. Caswell 2000). The function `anm.transM` provides an animated view of the population growth curves. The function `anm.TM.tck` provides a **tcltk** GUI to run `anm.TM.tck`.

Usage

```
transM(A, init, inter = 100, stage.names = c("All grps",1:(ncol(A))),
leg.room = 1.5, ...)

anm.transM(A,init,inter=100,stage.names =c("All grps",1:(ncol(A))),
leg.room = 1.5, anim.interval=0.1,...)

anm.TM.tck()
```

Arguments

`A`	Transition matrix containing survivorship probabilities and fecundities see Caswell (2000).
`init`	A numeric vector containing initial numbers in each age class of interest.
`inter`	Number of time intervals for which population numbers are to be calculated.
`stage.names`	A character vector giving life stage names.
`leg.room`	A Y-axis multiplier intended to create room for a legend.
`anim.interval`	Speed of animation in frames per second.
`...`	Additional arguments for `plot`

Value

Returns a plot and proportions of the population in each age class for the number of time intervals in `inter`.

Author(s)

Ken Aho

References

Caswell, H. 2000. *Matrix population models: Construction, analysis and interpretation, 2nd Edition*. Sinauer Associates, Sunderland, Massachusetts.

Gurevitch, J., Scheiner, S. M., and G. A. Fox. 2006. The ecology of Plants. Sinauer.

Examples

```
#Endangered cactus data data from Gurevitch et al. (2006)
A<-matrix(nrow=3,ncol=3,data=c(.672,0,.561,0.018,0.849,0,0,0.138,0.969),
byrow=TRUE)
init<-c(10,2,1)
transM(A,init,inter=100,stage.names=c("All","Sm. Juv.","Lg. Juv.","Adults"),
xlab="Years from present",ylab="n")
#animated version
## Not run:
anm.transM(A,init,inter=100,stage.names=c("All","Sm. Juv.","Lg. Juv.","Adults"),
xlab="Years from present",ylab="n")

## End(Not run)
```

trim.me *Trim data*

Description

Trims observations above and below the central $(1 - 2 \times \lambda)$ part of an an ordered vector of data.

Usage

```
trim.me(Y, trim = 0.2)
```

Arguments

`Y`	A vector of quantitative data.
`trim`	Proportion (0-1) to be trimmed from each tail of an ordered version of `Y`.

Value

Returns a trimmed data vector.

Author(s)

Ken Aho

Examples

```
x<-c(2,1,4,5,6,2.4,7,2.2,.002,15,17,.001)
trim.me(x)
```

trim.ranef.test *Robust test for random factors using trimmed means.*

Description

Provides a robust hypothesis test for the null: Var(X = 0), for a population of random factor levels.

Usage

```
trim.ranef.test(Y, X, tr = 0.2)
```

Arguments

`Y`	Vector of response data. A quantitative vector.
`X`	Vector of factor levels
`tr`	Amount of trimming. A number from 0-0.5.

Details

Robust analyses for random effect designs are particularly important since standard random effects models provide poor control over type I error when assumptions of normality and homoscedasticity are violated. Specifically, Wilcox (1994) showed that even with equal sample sizes, and moderately large samples, actual probability of type I error can exceed 0.3 if normality and homoscedasticity are violated.

Value

Returns a list with three components dataframe describing numerator and denominator degrees of freedom, the *F* test statistic and the *p*-value.

Note

code based on Wilcox (2005)

Author(s)

Ken Aho

References

Wilcox, R. R. 2005. *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

Examples

```
rye<-c(50,49.8,52.3,44.5,62.3,74.8,72.5,80.2,47.6,39.5,47.7,50.7)
nutrient<-factor(c(rep(1,4),rep(2,4),rep(3,4)))
trim.ranef.test(rye,nutrient,tr=.2)
```

---

trim.test *Robust one way trimmed means test.*

Description

A robust heteroscedastic procedure using trimmed means.

Usage

```
trim.test(Y, X, tr = 0.2)
```

Arguments

Y	A vector of responses. A quantitative vector
X	A vector of factor levels.
tr	The degree of trimming. A value from 0-0.5.

Details

The method utilized here is based on the simple idea of replacing means with trimmed means and standard error estimates, based on all the data, with the standard error of the trimmed mean (Wilcox 2005). The method has the additional benefit of being resistant to heteroscedasticity due to the use of the Welch method for calculating degrees of freedom. With no trimming the degrees of freedom reduce to those of the one way Welch procedure in `oneway.test`.

Value

Returns a dataframe with numerator and denominator degrees of freedom, a test statistic, and a *p*-value based on the *F*-distribution.

Note

code based on Wilcox (2005)

Author(s)

Ken Aho

References

Wilcox, R. R. 2005. *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

See Also

`oneway.test`

Examples

```
rye<-c(50,49.8,52.3,44.5,62.3,74.8,72.5,80.2,47.6,39.5,47.7,50.7)
nutrient<-factor(c(rep(1,4),rep(2,4),rep(3,4)))
trim.test(rye,nutrient,tr=.2)
```

`tukey.add.test` *Tukey's test of additivity.*

Description

With an RBD we are testing the null hypothesis that there is no treatment effect in any block. As a result randomized block designs including RBDs, Latin Squares, and spherical repeated measures assume that there is no interaction effect between blocks and main factors (i.e. main effects and block are additive). We can test this assumption with the Tukey's test for additivity. We address the following hypotheses:

H_0: Main effects and blocks are additive, versus H_A: Main effects and blocks are non-additive.

Usage

```
tukey.add.test(y, A, B)
```

Arguments

y	Response variable. Vector of quantitative data.
A	Main effects. Generally a vector of categorical data.
B	Blocking variable. A vector of categories (blocks).

Details

Tukey's test for additivity is best for detecting simple block x treatment interactions; for instance, when lines in an interaction plot cross. As a result interaction plots should be used for diagnosis of other types of interactions. A high probability of type II error results from the inability Tukey's additivity test to detect complex interactions (Kirk 1995). As a result a conservative value of should be used, i.e. 0.1 - 0.25.

Value

Returns a table with test results.

Author(s)

Orginal author unknown. Modified by K. Aho

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and W. Li. (2005) *Applied linear statistical models, 5th edition*. McGraw-Hill, Boston.

Kirk, R. E. 1995. *Experimental design*. Brooks/Cole. Pacific Grove, CA.

Examples

```
treatment<-as.factor(c(36,54,72,108,144,36,54,72,108,144,36,54,72,108,144))
block<-as.factor(c(rep(1,5),rep(2,5),rep(3,5)))
strength<-c(7.62,8.14,7.76,7.17,7.46,8,8.15,7.73,7.57,7.68, 7.93,7.87,7.74, 7.8,7.21)
tukey.add.test(strength,treatment,block)
```

V.mat *Pooled covariance matrices for multivariate procedures.*

Description

Calculates a pooled covariance matrix ala Legendre and Legendre (1998). This is required for a number of multivariate procedures including the Kullback statistic for multivariate homoscedasticity, and Hotelling's test.

Usage

```
V.mat(Y, X)
```

Arguments

Y	An n x p matrix of quantitative dependent variables.
X	A n x 1 of categorical groups (e.g. factor levels).

Author(s)

Ken Aho

References

Legendre, P, and L. Legendre (1998) *Numerical ecology, 2nd English edition*. Elsevier, Amsterdam, The Netherlands.

See Also

Kullback, Hotelling

Examples

```
Y1<-rnorm(100,12,2)
Y2<-rnorm(100,14,2)
X<-c(rep(1,50),rep(2,50))
V.mat(cbind(Y1,Y2),X)
```

`veg.table` *Summarized constancy/cover vegetation tables (not vegetables)*

Description

Calculates vegetation cover and constancy within groups (factor) for species at a defined minimum constancy,then converts constancy and cover to codes. Three different cover classes systems can be specified. A method developed by Aho (2006), Daubenmire cover classes, and Braun-Blanquet cover classes.

Usage

```
veg.table(Y, cat, min.const = 0.3, method = "aho", cover.only = FALSE)
```

Arguments

`Y`	A site x species (n x p) community matrix.
`cat`	An n x 1 vector of categorical assignments. Must be a factor.
`min.const`	A constancy cutoff. To help distill information data will be subset into two components species with >= `min.const` in at least one group, and < `min.const` across all groups. `min.const`=0.
`method`	The type of cover codes to use. The choices are possible `aho`, `daub`, and `braun`
`cover.only`	Logical. If only cover output is desired then stipulate `cover.only = TRUE`

Details

The function assumes that responses in the community matrix are percent cover, i.e. responses are generally in the range 0 - 100 (although responses greater than 100 are allowed). For constancy: 0 percent = ".", 0-10 = +, 10-20 = 1, 20-30 = 2, 30-40 = 3, 40-50 = 4, 50-60 = 5, 60-70 = 6,70-80 = 7, 80-90 = 8, 90-100 = 9.

Three different cover class systems can be specified.

Aho, i.e. method = `aho` cover classes use the following codes: 0 percent = ".", 0-0.01 = +, 0.01-1 = A, 1-2 = B, 2-5 = C, 5-25 = D, >25 = E.

Daubenmire,i.e. `daub` cover classes are as follows: 0 percent = ".", 0-5 percent = 1, 5-25 = 2, 25-50 = 3, 50-75 = 4, 75-95 = 5, >95 = 6.

Braun-Blanquet, i.e. `braun` cover classes are as follows: 0 percent = ".", 0-0.1 = "r", 0.1-1 = "+", 1-5 = 1, 5-25 = 2, 25-50 = 3, 50-75 = 4, >75 = 5.

The argument `min.const` allows creation of two summary cover and constancy matrices. The first contains species which have >= `min.const` within at least one of the categories/clusters. The second contains cover and constancy summaries for all other species

Value

A list with two of three possible components is returned. The components will change depending what is specified in `cover.only`

`const.cover` A matrix with constancy and cover ciphers containing species with >= 30 percent constancy in at least one group

`const.cover_less.than.min.const`
A matrix with constancy and cover ciphers containing species which do not have >= 30 percent constancy in at least one group

`cover` A matrix with constancy and cover ciphers containing species with >= 30 percent constancy in at least one group

Author(s)

Ken Aho; Tad Weaver contributed many of the ideas for cover/constancy ciphers and code ranges.

References

Aho, K. (2006) *Alpine ecology and subalpine cliff ecology in the Northern Rocky Mountains*. Doctoral dissertation, Montana State University, 458 pgs.

Gurevitch, J., Scheiner, S. M., and Fox, G. A. (2006) *The ecology of Plants*. Sinauer.

See Also

`const`

Examples

```
data(dune)
data(dune.env)
veg.table(Y=dune,cat=dune.env[,3])
```

`Venn` *Venn probability diagrams for an event with two outcomes*

Description

The user specifies the probabilities of two outcomes, and if applicable, their intersection. A Venn diagram is returned. The universe, S, will generally not have unit area, but in many applications will be a good approximation. The area of the intersection will also be an approximation.

Usage

```
Venn(A, B, AandB = 0, labA = "A", labB = "B")

Venn.tck()
```

Arguments

`A`	probability of event A
`B`	probability of event B
`AandB`	probability of the intersection of A and B
`labA`	Label assigned to event A in the diagram
`labB`	Label assigned to event B in the diagram

Value

A Venn diagram is returned.

Author(s)

K. Aho

References

Bain, L. J., and M. Engelhardt (1992) *Introduction to probability and mathematical statistics*. Duxbury press. Belmont, CA, USA.

Examples

```
Venn(A=.3,B=.2,AandB=.06)
```

`webs` *Spider web length data*

Description

Gosline et al. applied heat to strands of spider web to determine whether the structure underlying webs was rubber-like. Data are estimated from a graph in Gosline et al. (1984).

Usage

```
data(webs)
```

Format

The dataframe contains 4 columns

obs Observation number.

relative length Relative strand strand length after heat application.

temp.C Temp in degrees celsius.

residuals Residuals from the linear model length~temp.C.

Source

Gosline, J. M., Denny, M. W. and Demont, M. E. (1984). Spider silk as rubber. *Nature* 309, 551-552.

`wheat` *Agricultural randomized block design*

Description

Allard (1966) sought to quantify variation in the yield in wheat grasses. Five wheat crosses were selected from a breeding program and were grown at four randomly selected locations where the wheat would be grown commercially. At each location crosses (families) were randomly assigned to particular sections of fields, i.e. at each location a one way randomized block design was conducted.

Usage

```
data(wheat)
```

Format

The dataframe has four columns:

loc Refers to randomly selected locations where wheat were grown commercially. A factor with four levels: `1,2,3,4`.

block Refers to the replicate block within location. A factor with four levels: `1,2,3,4`. Within each block five wheat crosses were randomly assigned and grown.

fam Refers to wheat crosses. A factor with five levels: `1,2,3,4,5`.

yield Refers to wheat yield.

Source

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and O. Schabenberger. 2006. *SAS for mixed models 2nd ed.* SAS press.

win — *Winsorize data*

Description

Winsorizes the proportion of ordered data given by `lambda` from each tail.

Usage

```
win(x, lambda = 0.2)
```

Arguments

`x`	A vector of data.
`lambda`	A proportion from 0-1 giving the amount of data to be Winsorized in each tail of an ordered dataset.

Details

In Winsorization we we replace responses that are not in the central $1 - 2 \times \lambda$ part of an ordered sample with the minimum and maximum responses of the central part of the sample.

Value

Returns Winsorized data.

Author(s)

Ken Aho

References

Wilcox, R. R. (2005) *Introduction to Robust Estimation and Hypothesis Testing, Second Edition*. Elsevier, Burlington, MA.

Examples

```
x<-c(2,1,4,5,6,2.4,7,2.2,.002,15,17,.001)
win(x)
```

`world.co2` *World CO2 levels, by country, from 1980 to 2006*

Description

The US department of energy has compiled data since 1980 detailing total carbon dioxide emissions from the consumption and flaring of fossil fuels (in millions of metric tons of carbon dioxide). Data can be broken down by country.

Usage

```
data(world.co2)
```

Format

The dataframe contains 16 columns

Year The year of CO_2 measure (1980-2006).

Afghanistan CO_2 emissions in Afghanistan from 1980-2006 (1x10^6 metric tons).

Belgium CO_2 emissions in Belgium...

Brazil CO_2 emissions in Brazil...

Canada CO_2 emissions in Canada...

China CO_2 emissions in China...

Finland CO_2 emissions in Finland...

Ghana CO_2 emissions in Ghana...

Italy CO_2 emissions in Italy...

Japan CO_2 emissions in Japan...

Kenya CO_2 emissions in Kenya...

Mexico CO_2 emissions in Mexico...

Saudi.Arabia CO_2 emissions in Saudi Arabia...

United.Arab.Emirates CO_2 emissions in the United Arab Emirates..

United.States CO_2 emissions in United States...

World.Total CO_2 emissions totals for the world...

Source

US Department of energy; www.eia.doe.gov/emeu/iea/

`world.pop` *Population levels in various countries since 1980*

Description

Population levels of 13 countries since 1980. Population numbers are rounded to the nearest 100,000.

Usage

```
data(world.pop)
```

Format

The dataframe contains 14 columns

Year The year of population measurements (1980-2006)

Afghanistan Population in Afghanistan from 1980-2006, rounded to the nearest 100,000.

Brazil Population in Brazil...

Canada Population in Canada...

China Population in China...

Finland Population in Finland...

Italy Population in Italy...

Japan Population in Japan...

Kenya Population in Kenya...

Mexico Population in Mexico...

Saudi.Arabia Population in Saudi Arabia...

United.Arab.Emirates Population in the United Arab Emirates...

United.States Population in United States...

World.Total Population totals for the world...

Source

US census bureau; `http://www.census.gov/ipc/www/idb/idbsprd.html`

Index