Package 'spatialEco'

October 30, 2015

Type Package

Title Spatial Analysis and Modelling

Version 0.1-3

Date 2015-10-29

Description Utilities to support spatial data manipulation, query, sampling and modelling. Functions include models for species population density, download utilities for climate and global deforestation spatial products, spatial smoothing, multivariate separability, point process model for creating pseudo-absences and sub-sampling, polygon and point-distance landscape metrics, auto-logistic model, sampling models, cluster optimization and statistical exploratory tools.

Depends R (>= 3.2.1)

Imports sp (>= 1.1-1), raster (>= 2.4-15), spatstat (>= 1.35-0), cluster (>= 2.0.2), spdep (>= 0.5-88), SDMTools (>= 1.1-221), RCurl (>= 1.95-4.7), rgeos (>= 0.3-11), RANN (>= 2.5), rms (>= 4.3-1), yaImpute (>= 1.0-26), methods

Maintainer Jeffrey S. Evans <jeffrey_evans@tnc.org>

License GPL-3

URL http://cran.r-project.org/package=spatialEco

NeedsCompilation no

Repository CRAN

LazyData true

RoxygenNote 4.1.1.9001

Author Jeffrey S. Evans [aut, cre], Karthik Ram [ctb]

Date/Publication 2015-10-30 01:08:53

R topics documented:

`ants` *Ant Biodiversity Data*

Description

Roth et al., (1994) Costa Rican ant diversity data

Format

A data.frame with 82 rows (species) and 5 columns (covertypes):

species Ant species (family)

Primary.Forest Primary forest type

Abandoned.cacao.plantations Abandoned cacao plantations type

Productive.cacao.plantations Active cacao plantations type

Banana.plantations Active banana plantations type

Source

`http://www.tiem.utk.edu/~gross/bioed/bealsmodules/shannonDI.html`

References

Roth, D. S., I. Perfecto, and B. Rathcke (1994) The effects of management systems on ground-foraging ant diversity in Costa Rica. Ecological Applications 4(3):423-436.

`bearing.distance` *Bearing and Distance*

Description

Calculates a new point [X,Y] based on defined bearing and distance

Usage

```
bearing.distance(x, y, distance, azimuth, EastOfNorth = TRUE)
```

Arguments

`x`	x coordinate
`y`	y coordinate
`distance`	Distance to new point (in same units as x,y)
`azimuth`	Azimuth to new point
`EastOfNorth`	Specified surveying convention

Note

East of north is a surveying convention and defaults to true.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
pt <- cbind( x=480933, y=4479433)
bearing.distance(pt[1], pt[2], 1000, 40)
```

`breeding.density` *Breeding density areas*

Description

Calculates breeding density areas base on population counts/density and spatial point density.

Usage

```
breeding.density(x, pop, p = 0.75, bw = 1000, b = 6400, self = TRUE)
```

Arguments

x	sp SpatialPointsDataFrame object
pop	Population count/density column in x@data
p	Target percent of population
bw	Bandwidth distance
b	Buffer distance (eg., p < 0.75 b=6400m, p >= 0.75 b=8500m)
self	Should source observations be included in neighbour count (TRUE/FALSE)

Value

A list object with:

pop.pts sp point object with points identified within the specified p

pop.area sp polygon object of buffered points specified by parameter b

bandwidth Specified distance bandwidth used in identifying neighbour counts

buffer Specified buffer distance used in buffering points for pop.area

p Specified population percent

Note

The breeding density areas model identifies the Nth-percent population exhibiting the highest spatial density and counts/frequency. It then buffers these points by a specified distance to produce breeding area polygons.

depends: sp, rgeos

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Doherty, K.E., J.D. Tack, J.S. Evans, D.E. Naugle (2010) Mapping breeding densities of greater sage-grouse: A tool for range-wide conservation planning. Bureau of Land Management. Report Number L10PG00911

Examples

```
require(sp)
n=1500
bb <- rbind(c(-1281299,-761876.5),c(1915337,2566433.5))
  bb.mat <- cbind(c(bb[1,1], bb[1,2], bb[1,2], bb[1,1]),
                  c(bb[2,1], bb[2,1], bb[2,2], bb[2,2]))
    bbp <- Polygon(bb.mat)
     s <- spsample(bbp, n, type='random')
        pop <- SpatialPointsDataFrame(s, data.frame(ID=1:length(s),
                                counts=runif(length(s), 1,250)))

    bd75 <- breeding.density(pop, pop='counts', p=0.75, b=6400, bw=4000)
     plot(bd75$pop.area, main='75% breeding density areas')
       plot(pop, pch=20, col='black', add=TRUE)
         plot(bd75$pop.pts, pch=20, col='red', add=TRUE)
```

---

concordance *Concordance test for binomial models*

Description

Performs a concordance/disconcordance (C-statistic) test on binomial models.

Usage

```
concordance(y, p)
```

Arguments

y	vector of binomial response variable used in model
p	estimated probabilities from fit binomial model

Value

A list class object with the following components:

concordance percent of positives that are greater than probabilities of nulls

isconcordance inverse of concordance representing null class

tied number of tied probabilities

pairs number of pairs compared

Note

The C-statistic has been show to be comparable to the area under an ROC.

Test of binomial regression for the hypothesis that probabilities of all positives [1], are greater than the probabilities of the nulls [0]. The concordance would be 100

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Austin, P.C. & E.W. Steyerberg (2012) Interpreting the concordance statistic of a logistic regression model: relation to the variance and odds ratio of a continuous explanatory variable. BMC Medical Research Methodology, 12:82 Harrell, F.E. (2001) Regression modelling strategies. Springer, New York, NY. Royston, P. & D.G. Altman (2010) Visualizing and assessing discrimination in the logistic regression model. Statistics in Medicine 29(24):2508-2520

Examples

```
data(mtcars)
dat <- subset(mtcars, select=c(mpg, am, vs))
glm.reg <- glm(vs ~ mpg, data = dat, family = binomial)
concordance(dat$vs, predict(glm.reg, type = "response"))
```

---

conf.interval *Confidence interval for mean or median*

Description

Calculates confidence interval for the mean or median of a distribution with unknown population variance

Usage

```
conf.interval(x, cl = 0.95, stat = "mean", std.error = TRUE)
```

Arguments

x	Vector to calculate confidence interval for
cl	Percent confidence level (default = 0.95)
stat	Statistic (mean or median)
std.error	Return standard error (TRUE/FALSE)

Value

lci Lower confidence interval value

uci Upper confidence interval value

mean If stat = "mean", mean value of distribution

mean Value of the mean or median

conf.level Confidence level used for confidence interval

std.error If std.error = TRUE standard error of distribution

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
x <- runif(100)
cr <- conf.interval(x, cl = 0.97)
print(cr)

d <- density(x)
plot(d, type="n", main = "PDF with mean and 0.97 confidence interval")
  polygon(d, col="cyan3")
  abline(v=mean(x, na.rm = TRUE), lty = 2)
  segments( x0=cr[["lci"]], y0=mean(d$y), x1=cr[["uci"]],
            y1=mean(d$y), lwd = 2.5,
            col = "black")
 legend("topright", legend = c("mean", "CI"),
        lty = c(2,1), lwd = c(1,2.5))
```

---

correlogram *Correlogram*

Description

Calculates and plots a correlogram

Usage

```
correlogram(x, v, dist = 5000, dmatrix = FALSE, ns = 99,
  latlong = FALSE, ...)
```

Arguments

x	SpatialPointsDataFrame object
v	Test variable in x@data
dist	Distance of correlation lags, if latlong=TRUE units are in kilometres
dmatrix	Should the distance matrix be include in output (TRUE/FALSE)
ns	Number of simulations to derive simulation envelope
latlong	Coordinates are in latlong (TRUE/FALSE)
...	Arguments passed to cor ('pearson', 'kendall' or 'spearman')

Value

A list object containing:

autocorrelation is a data.frame object with the following components

autocorrelation - Autocorrelation value for each distance lag

dist - Value of distance lag

lci - Lower confidence interval (p=0.025)

uci - Upper confidence interval (p=0.975)

CorrPlot recordedplot object to recall plot

dmatrix Distance matrix (if dmatrix=TRUE)

Note

depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
library(sp)
data(meuse)
coordinates(meuse) = ~x+y
zinc.cg <- correlogram(x = meuse, v = meuse@data[,'zinc'], dist = 250, ns = 9)
```

csi *Cosine Similarity Index*

Description

Calculates the cosine similarity and angular similarity on two vectors or a matrix

Usage

```
csi(x, y = NULL)
```

Arguments

x	A vector or matrix object
y	If x is a vector, then a vector object

Value

If x is a matrix, a list object with: similarity and angular.similarity matrices

If x and y are vectors, a vector of similarity and angular.similarity

Note

The cosine similarity index is a measure of similarity between two vectors of an inner product space. This index is bested suited for high-dimensional positive variable space. One useful application of the index is to measure separability of clusters derived from algorithmic approaches (e.g., k-means). It is a good common practice to centre the data before calculating the index. It should be noted that the cosine similarity index is mathematically, and often numerically, equivalent to the Pearson's correlation coefficient

cosine similarity index is derived:

s(xy) = x * y / ||x|| * ||y||

expected 1.0 (perfect similarity) to -1.0 (perfect dissimilarity)

A normalised angle between the vectors can be used as a bounded similarity function within [0,1]

angular similarity = 1 - (cos(s)^-1/pi)

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
# Compare two vectors (centred using scale)
  x=runif(100)
  y=runif(100)^2
  csi(as.vector(scale(x)),as.vector(scale(y)))

  #' # Compare columns (vectors) in a matrix (centred using scale)
  x <- matrix(round(runif(100),0),nrow=20,ncol=5)
  ( s <- csi(scale(x)) )

# Compare vector (x) to each column in a matrix (y)
y <- matrix(round(runif(500),3),nrow=100,ncol=5)
x=runif(100)
csi(as.vector(scale(x)),scale(y))
```

---

dispersion *Dispersion (H-prime)*

Description

Calculates the dispersion ("rarity") of targets associated with planning units

Usage

```
dispersion(x)
```

Arguments

x	data.frame object of target values

Value

data.frame with columns H values for each target, H , sH, sHmax

Note

The dispersion index (H-prime) is calculated H = sum(sqrt(p) / sqrt(a)) where; P = [sum of target in planning unit / sum of target across all planning units] and a = [count of planning units containing target / number of planning units]

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Evans, J.S., S.R. Schill, G.T. Raber (2015) A Systematic Framework for Spatial Conservation Planning and Ecological Priority Design in St. Lucia, Eastern Caribbean. Chapter 26 in Central American Biodiversity : Conservation, Ecology and a Sustainable Future. F. Huettman (eds). Springer, NY.

Examples

```
 library(sp)
 data(pu)

 d <- dispersion(pu@data[,2:ncol(pu)])

## Not run:
 p <- d[,"H"]
 clr <- c("#3288BD", "#99D594", "#E6F598", "#FEE08B",
          "#FC8D59", "#D53E4F")
 clrs <- ifelse(p < 0.5524462, clr[1],
           ifelse(p >= 0.5524462 & p < 1.223523, clr[2],
             ifelse(p >= 1.223523 & p < 2.465613, clr[3],
            ifelse(p >= 2.465613 & p < 4.76429, clr[4],
              ifelse(p >= 4.76429 & p < 8.817699, clr[5],
                ifelse(p >= 8.817699, clr[6], NA))))))
 plot(pu, col=clrs, border=NA)
   legend("topleft", legend=rev(c("Very Rare","Rare","Moderately Rare",
          "Somewhat Common","Common","Over Dispersed")),
          fill=clr, cex=0.6, bty="n")
   box()

## End(Not run)
```

`download.daymet` *Download DAYMET*

Description

Batch download of daily gridded DAYMET climate data

Usage

```
download.daymet(years, tile, data.type = "all", download.folder = getwd(),
  http = "http://daymet.ornl.gov/thredds/fileServer/ornldaac/1219/tiles")
```

Arguments

`years`	Years to download (valid years 1980-2012)
`tile`	Tile index value (see url for tile index grid in notes section)

`data.type`	Type of climate metric: 'all', 'vp', 'tmin', 'tmax', 'swe', 'srad', 'prcp', 'dayl'.
`download.folder`	local download directory, defaults to current working directory
`http`	option to change URL

Value

DAYMET netCDF format climate metrics

Note

Available products

vp Water Vapour Pressure Daily average partial pressure of water vapour

tmin Daily minimum (degrees C) 2-meter air temperature

tmax Daily maximum (degrees C) 2-meter air temperature

swe Snow water equivalent (kg/m^2). Amount of water contained within snowpack.

srad Incident shortwave radiation flux density (W/m^2), taken as average over daylight period of the day.

prcp Daily total precipitation(mm/day), sum of all forms converted to water-equivalent.

dayl Duration of the daylight period for the day (s/day). Calculation is based on the period of the day during which the sun is above a hypothetical flat horizon.

DAYMET main website: http://daymet.ornl.gov

Tile index information http://daymet.ornl.gov/datasupport.html

Data respotory url: http://daymet.ornl.gov/thredds/fileServer/ornldaac/1219/tiles

Path structure: /year/tile_year/file.nc

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Thornton P.E., S.W. Running and M.A. White (1997) Generating surfaces of daily meteorological variables over large regions of complex terrain. Journal of Hydrology 190: 214-251.

Thornton, P.E. and S.W. Running (1999) An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agriculture and Forest Meteorology. 93:211-228.

Thornton, P.E., H. Hasenauer and M.A. White (2000) Simultaneous estimation of daily solar radiation and humidity from observed temperature and precipitation: An application over complex terrain in Austria. Agricultural and Forest Meteorology 104:255-271.

Examples

```
## Not run:
# Download 2009-2010 min and max temp for tiles 11737 and 11738
 laramie.plains <- c(11737, 11738)
 my.years <- c(seq(2009,2010,1))
 download.daymet(years=my.years, tile=laramie.plains, data.type=c('tmin','tmax'))

## End(Not run)
```

`download.hansen` *Download Hansen Forest 2000-2013 Change*

Description

Download of Hansen Global Forest Change 2000-2013

Usage

```
download.hansen(tile, data.type = c("loss"), download.folder = getwd())
```

Arguments

`tile`	Granule index (See project URL for granule grid index)
`data.type`	Type of data to download options: 'treecover2000', 'loss', 'gain', 'lossyear', 'datamask', 'first', 'last'
`download.folder`	Destination folder

Value

Downloaded Hansen forest loss tif files

Note

Project website with 10x10 degree granule index: http://earthenginepartners.appspot.com/science-2013-global-forest/download_v1.1.html

Avaliable products: treecover2000, loss, gain, lossyear, datamask, first, or last

'treecover2000' (Tree canopy cover for year 2000) - Tree cover in the year 2000, defined as canopy closure for all vegetation taller than 5m in height. Encoded as a percentage per output grid cell, in the range 0-100.

'loss' (Global forest cover loss 2000-2013) - Forest loss during the period 2000-2013, defined as a stand-replacement disturbance, or a change from a forest to non-forest state. Encoded as either 1 (loss) or 0 (no loss).

'gain' (Global forest cover gain 2000-2012) - Forest gain during the period 2000-2012, defined as the inverse of loss, or a non-forest to forest change entirely within the study period. Encoded as either 1 (gain) or 0 (no gain).

'lossyear' (Year of gross forest cover loss event) - A disaggregation of total forest loss to annual time scales. Encoded as either 0 (no loss) or else a value in the range 1-13, representing loss detected primarily in the year 2001-2013.

'datamask' (Data mask) - Three values representing areas of no data (0), mapped land surface (1), and permanent water bodies (2).

'first' (Circa year 2000 Landsat 7 cloud-free image composite) - Reference multispectral imagery from the first available year, typically 2000. If no cloud-free observations were available for year 2000, imagery was taken from the closest year with cloud-free data, within the range 1999-2012.

'last' (Circa year 2013 Landsat cloud-free image composite) - Reference multispectral imagery from the last available year, typically 2013. If no cloud-free observations were available for year 2013, imagery was taken from the closest year with cloud-free data, within the range 2010-2012.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Hansen, M. C., P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R. G. Townshend. (2013) High-Resolution Global Maps of 21st-Century Forest Cover Change. Science 342:850-53.

Examples

```
## Not run:
# Download single tile
 download.hansen(tile=c('00N', '130E'), data.type=c('loss', 'lossyear'),
                 download.folder=getwd())

# Batch download of multiple tiles
tiles <- list(c('00N', '140E'), c('00N', '130E'))
  for( j in 1:length(tiles)){
    download.hansen(tile=tiles[[j]], data.type=c('loss'))
 }

## End(Not run)
```

`download.prism` *Download PRISM*

Description

Batch download of monthly gridded PRISM climate data

Usage

```
download.prism(data.type, date.range, time.step = "monthly",
  download.folder = getwd(), by.year = FALSE, unzip.file = TRUE,
  ftp.site = "ftp://prism.oregonstate.edu")
```

Arguments

data.type	Specify climate metric ('ppt','tmin','tmax','tmean')
date.range	A vector with start and end date in y/m/d format
time.step	Timestep of product ('daily'/'monthly')
download.folder	Local download directory, defaults to current working directory
by.year	Create a directory for each year (TRUE/FALSE)
unzip.file	Unzip file on download (TRUE/FALSE)
ftp.site	PRISM ftp address to use, default: ftp://prism.oregonstate.edu

Value

Compressed or uncompressed PRISM monthly gridded data(bil raster format)

Note

Monthly data 1895-1980 is available in a single zip file on the ftp site

PRISM URL: http://prism.nacse.org/

FTP download sites for 400m gridded daily/monthly climate data

ftp://prism.oregonstate.edu/daily

ftp://prism.oregonstate.edu/monthly

Naming convention: PRISM_<var>_<stability>_<scale&version>_<date>_bil.zip

i.e., 'PRISM_ppt_stable_4kmD1_20100208_bil.zip'

Data description:

http://prism.nacse.org/documents/PRISM_datasets_aug2013.pdf

depends: RCurl

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
## Not run:
   my.dates <- c('2000/1/1', '2001/12/30')
   download.prism('ppt', date.range=my.dates, time.step='monthly', by.year=TRUE)

# Download monthly precipitation data Jan 1st 2000 to Feb 10th 2000 (n=41)
   my.dates <- c('2000/1/1', '2000/2/10')
```

```
  download.prism('ppt', date.range=my.dates, time.step='daily', by.year=TRUE)

## End(Not run)
```

`gaussian.kernel` *Gaussian Kernel*

Description

Creates a Gaussian Kernel of specified size and sigma

Usage

```
gaussian.kernel(sigma = 2, n = 5)
```

Arguments

`sigma`	sigma (standard deviation) of kernel (defaults 2)
`n`	size of symmetrical kernel (defaults to 5x5)

Value

matrix of gaussian distribution

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Riley, S.J., S.D. DeGloria and R. Elliot (1999) A terrain ruggedness index that quantifies topographic heterogeneity, Intermountain Journal of Sciences 5(1-4):23-27.

Examples

```
par(mfrow=c(2,2))
persp(gaussian.kernel(sigma=1, n=27), theta = 135, phi = 30, col = "grey",
        ltheta = -120, shade = 0.6, border=NA )
persp(gaussian.kernel(sigma=2, n=27), theta = 135, phi = 30, col = "grey",
      ltheta = -120, shade = 0.6, border=NA )
persp(gaussian.kernel(sigma=3, n=27), theta = 135, phi = 30, col = "grey",
      ltheta = -120, shade = 0.6, border=NA )
persp(gaussian.kernel(sigma=4, n=27), theta = 135, phi = 30, col = "grey",
      ltheta = -120, shade = 0.6, border=NA )
```

group.pdf *Probability density plot by group*

Description

Creates a probability density plot of y for each group of x

Usage

```
group.pdf(x, y, col = NULL, lty = NULL, lwd = NULL, lx = "topleft",
  ly = NULL, ...)
```

Arguments

x	Numeric, character or factorial vector of grouping variable (must be same length as y)
y	Numeric vector (density variable)
col	Optional line colors (see par, col)
lty	Optional line types (see par, lty)
lwd	Optional line widths (see par, lwd)
lx	Position of legend (x coordinate or 'topright', 'topleft', 'bottomright', 'bottomleft')
ly	Position of legend (y coordinate)
...	Additional agruments passed to plot

Author(s)

Jeffrey S. Evans <jeffrey_evans<at>tnc.org>

References

Simonoff, J. S. (1996). Smoothing Methods in Statistics. Springer-Verlag, New York.

Examples

```
y=dnorm(runif(100))
x=rep(c(1,2,3), length.out=length(y))
group.pdf(x=as.factor(x), y=y, main='Probability Density of y by group(x)',
ylab='PDF', xlab='Y', lty=c(1,2,3))
```

hexagons *Hexagons*

Description

Create hexagon polygons

Usage

```
hexagons(x, res = 100, ...)
```

Arguments

x	sp SpatialDataFrame class object
res	Area of resulting hexagons
...	Additional arguments passed to spsample

Value

SpatialPolygonsDataFrame OBJECT

Note

depends: sp

Examples

```
require(sp)
  data(meuse)
    coordinates(meuse) <- ~x+y

hex.polys <- hexagons(meuse, res=100)
  plot(hex.polys)
    plot(meuse,pch=20,add=TRUE)

# Points intersecting hexagons
hex.pts <- na.omit(over(meuse,hex.polys))
(hex.pts <- data.frame(PTID=rownames(hex.pts), hex.pts))
```

idw.smoothing *idw.smoothing*

Description

Distance weighted smoothing of a variable in a spatial point object

Usage

```
idw.smoothing(x, y, d, k)
```

Arguments

x	Object of class SpatialPointsDataFrame
y	Numeric data in x@data
d	Distance constraint
k	Maximum number of k-nearest neighbours within d

Value

A vector, same length as nrow(x), of adjusted y values

Note

Smoothing is conducted with a weighted-mean where; weights represent inverse standardized distance lags Distance-based or neighbour-based smoothing can be specified by setting the desired neighbour smoothing method to a specified value then the other parameter to the potential maximum. For example; a constraint distance, including all neighbours within 1000 (d=1000) would require k to equal all of the potential neighbours (n-1 or k=nrow(x)-1).

Depends: sp, RANN

Examples

```
 library(sp)
  data(meuse)
  coordinates(meuse) <- ~x+y

# Calculate distance weighted mean on cadmium variable in meuse data
  cadmium.idw <- idw.smoothing(meuse, 'cadmium', k=nrow(meuse), d = 1000)
  meuse@data$cadmium.wm <- cadmium.idw
  opar <- par
  par(mfrow=c(2,1))
  plot(density(meuse@data$cadmium), main='Cadmium')
  plot(density(meuse@data$cadmium.wm), main='IDW Cadmium')
  par <- opar
```

`insert.values` *Insert Values*

Description

Inserts new values into a vector at specified positions

Usage

```
insert.values(x, y, l)
```

Arguments

x	A vector to insert values
y	Values to insert into x
l	Index position(s) to insert y values into x

Value

A vector with values of y inserted into x

Note

This function inserts new values at specified positions in a vector. It does not replace existing values. If a single value is provided for y and l represents multiple positions y will be replicated for the length of l. In this way you can insert the same value at multiple locations.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
(x=1:10)

# Insert single value in one location
insert.values(x, 100, 2)

# Insert multiple values in multiple locations
insert.values(x, c(100,200), c(2,8))

# Insert single value in multiple locations
insert.values(x, NA, c(2,8))
```

`kl.divergence` *Kullback-Leibler divergence (relative entropy)*

Description

Calculates the Kullback-Leibler divergence (relative entropy) between unweighted theoretical component distributions. Divergence is calculated as: int [f(x) (log f(x) - log g(x)) dx] for distributions with densities f() and g().

Usage

```
kl.divergence(object, eps = 10^-4, overlap = TRUE)
```

Arguments

`object`	Matrix or dataframe object with >=2 columns
`eps`	Probabilities below this threshold are replaced by this threshold for numerical stability.
`overlap`	Logical, do not determine the KL divergence for those pairs where for each point at least one of the densities has a value smaller than eps.

Value

pairwise Kullback-Leibler divergence index (matrix)

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Kullback S., and R. A. Leibler (1951) On information and sufficiency. The Annals of Mathematical Statistics 22(1):79-86

Examples

```
x <- seq(-3, 3, length=200)
y <- cbind(n=dnorm(x), t=dt(x, df=10))
  matplot(x, y, type='l')
    kl.divergence(y)

# extract value for last column
  kl.divergence(y[,1:2])[3:3]
```

`land.metrics` *Landscape metrics for points and polygons*

Description

Calculates a variety of landscape metrics, on binary rasters, for polygons or points with a buffer distance

Usage

```
land.metrics(x, y, bkgd = NA, metrics = c(4, 14, 33, 34, 35, 37, 38),
  bw = 1000, latlon = FALSE, trace = TRUE)
```

Arguments

`x`	SpatalPointsDataFrame or SpatalPolgonsDataFrame class object
`y`	raster class object
`bkgd`	Background value (will be ignored)
`metrics`	Numeric index of desired metric (see available metrics)
`bw`	Buffer distance (ignored if x is SpatalPolgonsDataFrame)
`latlon`	Is raster data in lat-long (TRUE/FALSE)
`trace`	Plot raster subsets and echo object ID at each iteration (TRUE | FALSE)

Value

If multiple classes are evaluated a list object with a data.frame for each class contaning specified metrics in columns. The data.frame is ordered and shares the same row.names as the input feature class and can be directly joined to the @data slot. For single class problems a data.frame object is returned.

Note

[2]n.patches, [3]total.area, [4]prop.landscape [5]patch.density, [6]total.edge, [7]edge.density, [8]landscape.shape.index [9]largest.patch.index, [10]mean.patch.area, [11]sd.patch.area, [12]min.patch.area [13]max.patch.area, [14]perimeter.area.frac.dim, [15]mean.perim.area.ratio, [16]sd.perim.area.ratio [17]min.perim.area.ratio, [18]max.perim.area.ratio, [19]mean.shape.index, [20]sd.shape.index [21]min.shape.index, [22]max.shape.index, [23]mean.frac.dim.index, [24]sd.frac.dim.index [25]min.frac.dim.index, [26]max.frac.dim.index, [27]total.core.area, [28]prop.landscape.core [29]mean.patch.core.area, [30]sd.patch.core.area, [31]min.patch.core.area, [32]max.patch.core.area [33]prop.like.adjacencies, [34]aggregation.index, [35]lanscape.division.index, [36]splitting.index [37]effective.mesh.size, [38]patch.cohesion.index

Modifications to the function incorporate multi-class metrics by fetching the unique values of the raster and creating a list object contaning a data.frame for each class. Unfortunately, retrieving unique values is a very slow function.

depends: sp, raster, rgeos, SDMTools

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
library(raster)
library(sp)

r <- raster(nrows=180, ncols=360, xmn=571823.6, xmx=616763.6, ymn=4423540,
            ymx=4453690, resolution=270, crs = CRS("+proj=utm +zone=12 +datum=NAD83
            +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0"))

r[] <- rpois(ncell(r), lambda=1)
r <- calc(r, fun=function(x) { x[x >= 1] <- 1; return(x) } )
x <- sampleRandom(r, 10, na.rm = TRUE, sp = TRUE)

( class.1 <- land.metrics(x=x, y=r, bw=1000, bkgd = 0, metrics = c(4,7,33,34)) )
( all.class <- land.metrics(x=x, y=r, bw=1000, bkgd = NA, metrics = c(4,7,33,34)) )

# Pull metrics associated with class "0"
all.class[["0"]]
```

`local.min.max` *Local minimum and maximum*

Description

Calculates the local minimums and maximums in a numeric vector, indicating inflection points in the distribution.

Usage

```
local.min.max(x, dev = mean, plot = TRUE, add.points = FALSE, ...)
```

Arguments

`x`	A numeric vector
`dev`	Deviation statistic (mean or median)
`plot`	plot the minimum and maximum values with the distribution (TRUE/FALSE)
`add.points`	Should all points of x be added to plot (TRUE/FALSE)
`...`	Arguments passed to plot

Value

A list object with:

minima minimum local values of x

maxima maximum local values of x

mindev Absolute deviation of minimum from specified deviation statistic (dev argument)

maxdev Absolute deviation of maximum from specified deviation statistic (dev argument)

Note

Useful function for identifying inflection or enveloping points in a distribution

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
x <- rnorm(100,mean=1500,sd=800)
( lmm <- local.min.max(x, dev=mean, add.points=TRUE, main="Local Minima and Maxima") )

# return only local minimum values
   local.min.max(x)$minima
```

---

loess.boot *Loess Bootstrap*

Description

Bootstrap of a Local Polynomial Regression (loess)

Usage

```
loess.boot(x, y, nreps = 100, confidence = 0.95, ...)
```

Arguments

x	Independent variable
y	Dependent variable
nreps	Number of bootstrap replicates
confidence	Fraction of replicates contained in confidence region
...	Additional arguments passed to loess function

Value

nreps Number of bootstrap replicates

confidence Confidence interval (region)

span alpha (span) parameter used loess fit

degree polynomial degree used in loess fit

normalize Normalized data (TRUE/FALSE)

family Family of statistic used in fit

parametric Parametric approximation (TRUE/FALSE)

surface Surface fit, see loess.control

data data.frame of x,y used in model

fit data.frame including: x Equally-spaced x index (see NOTES) y.fit loess fit up.lim Upper confidence interval low.lim Lower confidence interval stddev Standard deviation of loess fit at each x value

Note

The function fits a loess curve and then calculates a symmetric nonparametric bootstrap with a confidence region. Fitted curves are evaluated at a fixed number of equally-spaced x values, regardless of the number of x values in the data. Some replicates do not include the values at the lower and upper end of the range of x values. If the number of such replicates is too large, it becomes impossible to construct a confidence region that includes a fraction "confidence" of the bootstrap replicates. In such cases, the left and/or right portion of the confidence region is truncated.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Cleveland, WS, (1979) Robust Locally Weighted Regression and Smoothing Plots Journal of the American Statistical Association 74:829-836

Efron, B., and R. Tibshirani (1993) An Introduction to the Bootstrap Chapman and Hall, New York

Hardle, W., (1989) Applied Nonparametric Regression Cambridge University Press, NY.

Tibshirani, R. (1988) Variance stabilization and the bootstrap. Biometrika 75(3):433-44.

Examples

```
n=1000
x <- seq(0, 4, length.out=n)
y <- sin(2*x)+ 0.5*x + rnorm(n, sd=0.5)
sb <- loess.boot(x, y, nreps=99, confidence=0.90, span=0.40)
plot(sb)
```

`loess.ci` *Loess with confidence intervals*

Description

Calculates a local polynomial regression fit with associated confidence intervals

Usage

```
loess.ci(y, x, p = 0.95, plot = FALSE, ...)
```

Arguments

`y`	Dependent variable, vector
`x`	Independent variable, vector
`p`	Percent confidence intervals (default is 0.95)
`plot`	Plot the fit and confidence intervals
`...`	Arguments passed to loess

Value

A list object with:

loess Predicted values

se Estimated standard error for each predicted value

lci Lower confidence interval

uci Upper confidence interval

df Estimated degrees of freedom

rs Residual scale of residuals used in computing the standard errors

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of Statistical Models in S eds J.M. Chambers and T.J. Hastie, Wadsworth & Brooks/Cole.

Examples

```
x <- seq(-20, 20, 0.1)
y <- sin(x)/x + rnorm(length(x), sd=0.03)
p <- which(y == "NaN")
  y <- y[-p]
  x <- x[-p]

par(mfrow=c(2,2))
lci <- loess.ci(y, x, plot=TRUE, span=0.10)
lci <- loess.ci(y, x, plot=TRUE, span=0.30)
lci <- loess.ci(y, x, plot=TRUE, span=0.50)
lci <- loess.ci(y, x, plot=TRUE, span=0.80)
```

---

logistic.regression *Logistic and Auto-logistic regression*

Description

Performs a logistic (binomial) or auto-logistic (spatially lagged binomial) regression using maximum likelihood or penalized maximum likelihood estimation.

Usage

```
logistic.regression(ldata, y, x, penalty = TRUE, autologistic = FALSE,
  coords = NULL, bw = NULL, type = "inverse", style = "W",
  longlat = FALSE, ...)
```

Arguments

ldata	data.frame object containing variables
y	Dependent variable (y) in ldata
x	Independent variable(s) (x) in ldata
penalty	Apply regression penalty (TRUE/FALSE)
autologistic	Add auto-logistic term (TRUE/FALSE)
coords	Geographic coordinates for auto-logistic model matrix or sp object.
bw	Distance bandwidth to calculate spatial lags (if empty neighbours result, need to increase bandwith). If not provided it will be calculated automatically based on the minimum distance that includes at least one neighbor.
type	Neighbour weighting scheme (see autocov_dist)
style	Type of neighbour matrix (Wij), default is mean of neighbours
longlat	Are coordinates (coords) in geographic, lat/long (TRUE/FALSE)
...	Additional arguments passed to lrm

Value

A list class object with the following components:

model lrm model object (rms class)

bandwidth If AutoCov = TRUE returns the distance bandwidth used for the auto-covariance function

diagTable data.frame of regression diagnostics

coefTable data.frame of regression coefficients

Residuals data.frame of residuals and standardized residuals

AutoCov If an auto-logistic model, AutoCov represents lagged auto-covariance term

Note

It should be noted that the auto-logistic model (Besag 1972) is intended for exploratory analysis of spatial effects. Auto-logistic are know to underestimate the effect of environmental variables and tend to be unreliable (Dormann 2007).

Wij matrix options under style argument - B is the basic binary coding, W is row standardised (sums over all links to n), C is globally standardised (sums over all links to n), U is equal to C divided by the number of neighbours (sums over all links to unity) and S is variance-stabilizing.

Spatially lagged y defined as:

W(y)ij=sumj_(Wij yj)/ sumj_(Wij)

where; Wij=1/Euclidean[i,j]

depends: rms, spdep

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Besag, J.E., (1972) Nearest-neighbour systems and the auto-logistic model for binary data. Journal of the Royal Statistical Society, Series B Methodological 34:75-83

Dormann, C.F., (2007) Assessing the validity of autologistic regression. Ecological Modelling 207:234-242

Le Cessie, S., Van Houwelingen, J.C., (1992) Ridge estimators in logistic regression. Applied Statistics 41:191-201

Shao, J., (1993) Linear model selection by cross-validation. JASA 88:486-494

Examples

```
require(sp)
require(spdep)
require(rms)
data(meuse)
  coordinates(meuse) <- ~x+y
    meuse@data <- data.frame(DepVar=rbinom(dim(meuse)[1], 1, 0.5), meuse@data)
```

```
#### Logistic model
lmodel <- logistic.regression(meuse@data, y='DepVar', x=c('dist','cadmium','copper'))
  lmodel$model
    lmodel$diagTable
      lmodel$coefTable

### Auto-logistic model using 'autocov_dist' in 'spdep' package
lmodel <- logistic.regression(meuse@data, y='DepVar', x=c('dist','cadmium','copper'),
                              autologistic=TRUE, coords=coordinates(meuse), bw=5000)
  lmodel$model
    lmodel$diagTable
      lmodel$coefTable
  est <- predict(lmodel$model, type='fitted.ind')

#### Add residuals, standardized residuals and estimated probabilities
VarNames <- rownames(lmodel$model$var)[-1]
  meuse@data$AutoCov <- lmodel$AutoCov
    meuse@data <- data.frame(meuse@data, Residual=lmodel$Residuals[,1],
                             StdResid=lmodel$Residuals[,2], Probs=predict(lmodel$model,
                             meuse@data[,VarNames],type='fitted') )

#### Plot fit and probabilities
resid(lmodel$model, "partial", pl="loess")
resid(lmodel$model, "partial", pl=TRUE)                  # plot residuals
resid(lmodel$model, "gof")                              # global test of goodness of fit
lp1 <- resid(lmodel$model, "lp1")                    # Approx. leave-out linear predictors
-2 * sum(meuse@data$DepVar * lp1 + log(1-plogis(lp1)))  # Approx leave-out-1 deviance
spplot(meuse, c('Probs'))                           # plot estimated probabilities at points
```

`moments` *moments*

Description

Calculate statistical moments of a distribution

Usage

```
moments(x, plot = FALSE)
```

Arguments

`x`	numeric vector
`plot`	plot of distribution (TRUE/FALSE)

Value

A vector with the following values

min Minimum

25th 25th percentile

mean Arithmetic mean

gmean Geometric mean

hmean Harmonic mean

median 50th percentile

7th5 75th percentile

max Maximum

stdv Standard deviation

var Variance

cv Coefficient of variation (percent)

mad Median absolute deviation

skew Skewness

kurt Kurtosis

nmodes Number of modes

mode Mode (dominate)

Author(s)

Jeffrey S. Evans <jeffrey_evans<at>tnc.org>

Examples

```
x <- runif(1000,0,100)
( d <- moments(x, plot=TRUE) )
( mode.x <- moments(x, plot=FALSE)[16] )
```

---

nni *Average Nearest Neighbour Index (NNI)*

Description

Calculates the NNI as a measure of clustering or dispersal

Usage

```
nni(x, win = "hull")
```

Arguments

x	An sp point object
win	Type of window 'hull' or 'extent'

Value

NNI value

Note

The nearest neighbour index is expressed as the ratio of the observed distance divided by the expected distance. The expected distance is the average distance between neighbours in a hypothetical random distribution. If the index is less than 1, the pattern exhibits clustering; if the index is greater than 1, the trend is toward dispersion or competition. The Nearest Neighbour Index is calculated as: Nearest Neighbour Distance (observed) D(nn) = sum(min(Dij)/N) Mean Random Distance (expected) D(e) = 0.5 SQRT(A/N) Nearest Neighbour Index NNI = D(nn)/D(e) Where; D=neighbour distance, A=Area

Depends: sp, spatstat

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Clark, P.J., and F.C. Evans (1954) Distance to nearest neighbour as a measure of spatial relationships in populations. Ecology 35:445-453

Cressie, N (1991) Statistics for spatial data. Wiley & Sons, New York.

Examples

```
require(sp)
data(meuse)
  coordinates(meuse) <- ~x+y
    nni(meuse)
```

---

o.ring *Inhomogeneous O-ring*

Description

Calculates the inhomogeneous O-ring point pattern statistic (Wiegand & Maloney 2004)

Usage

```
o.ring(x, inhomogeneous = FALSE, ...)
```

Arguments

x	spatstat ppp object
inhomogeneous	(boolean) Run homogeneous (pcf) or inhomogeneous (pcfinhom)
...	additional arguments passed to pcf or pcfinhom

Value

plot of o-ring and data.frame with plot labels and descriptions

Note

The function K(r) is the expected number of points in a circle of radius r centred at an arbitrary point (which is not counted), divided by the intensity l of the pattern. The alternative pair correlation function g(r), which arises if the circles of Ripley's K-function are replaced by rings, gives the expected number of points at distance r from an arbitrary point, divided by the intensity of the pattern. Of special interest is to determine whether a pattern is random, clumped, or regular.

Using rings instead of circles has the advantage that one can isolate specific distance classes, whereas the cumulative K-function confounds effects at larger distances with effects at shorter distances. Note that the K-function and the O-ring statistic respond to slightly different biological questions. The accumulative K-function can detect aggregation or dispersion up to a given distance r and is therefore appropriate if the process in question (e.g., the negative effect of competition) may work only up to a certain distance, whereas the O-ring statistic can detect aggregation or dispersion at a given distance r. The O-ring statistic has the additional advantage that it is a probability density function (or a conditioned probability spectrum) with the interpretation of a neighborhood density, which is more intuitive than an accumulative measure.

Depends: spatstat

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Wiegand T., and K. A. Moloney (2004) Rings, circles and null-models for point pattern analysis in ecology. Oikos 104:209-229

Examples

```
library(spatstat)
data(lansing)
    x  <- spatstat::unmark(split(lansing)$maple)
    o.ring(x)
```

optimal.k *optimalK*

Description

Find optimal k of k-Medoid partitions using silhouette widths

Usage

```
optimal.k(x, nk = 10, plot = TRUE, cluster = TRUE, clara = FALSE, ...)
```

Arguments

x	Numeric dataframe, matrix or vector
nk	Number of clusters to test (2:nk)
plot	Plot cluster silhouettes(TRUE/FALSE)
cluster	Create cluster object with optimal k
clara	Use clara model for large data
...	Additional arguments passed to clara

Note

Depends: cluster

Author(s)

Jeffrey S. Evans <jeffrey_evans<at>tnc.org>

References

Theodoridis, S. & K. Koutroumbas(2006) Pattern Recognition 3rd ed.

Examples

```
require(cluster)
  x <- rbind(cbind(rnorm(10,0,0.5), rnorm(10,0,0.5)),
             cbind(rnorm(15,5,0.5), rnorm(15,5,0.5)))

  clust <- optimal.k(x, 20, plot=TRUE, cluster=TRUE)
    plot(silhouette(clust), col = c('red', 'green'))
      plot(clust, which.plots=1, main='K-Medoid fit')

# Extract multivariate and univariate mediods (class centres)
  clust$medoids
    pam(x[,1], 1)$medoids

# join clusters to data
  x <- data.frame(x, k=clust$clustering)
```

`outliers` *Outliers*

Description

Identify outliers using modified Z-score

Usage

```
outliers(x)
```

Arguments

`x`	A numeric vector

Value

value for the modified Z-score

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Iglewicz, B. & D.C. Hoaglin (1993) How to Detect and Handle Outliers, American Society for Quality Control, Milwaukee, WI.

Examples

```
# Create data with 3 outliers
   x <- seq(0.1, 5, length=100)
   x[98:100] <- c(100, 55, 250)

# Calculate Z score
    Z <- outliers(x)

# Show number of extreme outliers using Z-score
    length(Z[Z > 9.9])

# Remove extreme outliers
    x <- x[-which(Z > 9.9)]
```

parea.sample *Percent area sample*

Description

Creates a point sample of polygons where n is based on percent area

Usage

```
parea.sample(x, pct = 0.1, join = FALSE, msamp = 1, sf = 4046.86,
  stype = "hexagonal", ...)
```

Arguments

x	sp SpatialPolygonsDataFrame object
pct	Percent of area sampled
join	Join polygon attributed to point sample
msamp	Minimum samples
sf	Scaling factor (default is meters to acres conversion factor)
stype	Sampling type ('random', 'regular', 'nonaligned', 'hexagonal')
...	Additional arguments passed to spsample

Value

A SpatialPointsDataFrame with polygon samples

Note

This function results in an adaptive sample based on the area of each polygon

Depends: sp,

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
require(sp)
sr1=Polygons(list(Polygon(cbind(c(180114, 180553, 181127, 181477, 181294, 181007, 180409,
  180162, 180114), c(332349, 332057, 332342, 333250, 333558, 333676,
  332618, 332413, 332349)))),'1')
sr2=Polygons(list(Polygon(cbind(c(180042, 180545, 180553, 180314, 179955, 179142, 179437,
  179524, 179979, 180042), c(332373, 332026, 331426, 330889, 330683,
  331133, 331623, 332152, 332357, 332373)))),'2')
sr=SpatialPolygons(list(sr1,sr2))
srdf=SpatialPolygonsDataFrame(sr, data.frame(row.names=c('1','2'), PIDS=1:2))
```

```
ars <- parea.sample(srdf, pct=0.20, stype='random')
  plot(srdf)
    plot(ars, pch=20, add=TRUE)
```

`plot.loess.boot` *Plot Loess Bootstrap*

Description

Plot function for loess.boot object

Usage

```
## S3 method for class 'loess.boot'
plot(x, ...)
```

Arguments

`x`	A loess.boot object
`...`	Additional arguments passed to plot

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Cleveland, WS, (1979) Robust Locally Weighted Regression and Smoothing Plots Journal of the American Statistical Association 74:829-836

Efron, B., and R. Tibshirani (1993) An Introduction to the Bootstrap Chapman and Hall, New York

Hardle, W., (1989) Applied Nonparametric Regression Cambridge University Press, NY.

Tibshirani, R. (1988) Variance stabilization and the bootstrap. Biometrika 75(3):433-44.

Examples

```
n=1000
x <- seq(0, 4, length.out=n)
y <- sin(2*x)+ 0.5*x + rnorm(n, sd=0.5)
sb <- loess.boot(x, y, nreps = 99, confidence = 0.90, span = 0.40)
plot(sb)
```

point.in.poly *Point and Polygon Intersect*

Description

Intersects point and polygon feature classes and adds polygon attributes to points

Usage

```
point.in.poly(pts, polys)
```

Arguments

pts	sp SpatialPointsDataFrame or SpatialPoints object
polys	sp SpatialPolygonsDataFrame object

Value

A SpatialPointsDataFrame with intersected polygon attributes

Note

Depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
require(sp)
data(meuse)
coordinates(meuse) = ~x+y
sr1=Polygons(list(Polygon(cbind(c(180114, 180553, 181127, 181477, 181294, 181007, 180409,
  180162, 180114), c(332349, 332057, 332342, 333250, 333558, 333676,
  332618, 332413, 332349)))),'1')
sr2=Polygons(list(Polygon(cbind(c(180042, 180545, 180553, 180314, 179955, 179142, 179437,
  179524, 179979, 180042), c(332373, 332026, 331426, 330889, 330683,
  331133, 331623, 332152, 332357, 332373)))),'2')
sr3=Polygons(list(Polygon(cbind(c(179110, 179907, 180433, 180712, 180752, 180329, 179875,
  179668, 179572, 179269, 178879, 178600, 178544, 179046, 179110),
  c(331086, 330620, 330494, 330265, 330075, 330233, 330336, 330004,
  329783, 329665, 329720, 329933, 330478, 331062, 331086)))),'3')
sr4=Polygons(list(Polygon(cbind(c(180304, 180403,179632,179420,180304),
  c(332791, 333204, 333635, 333058, 332791)))),'4')
sr=SpatialPolygons(list(sr1,sr2,sr3,sr4))
srdf=SpatialPolygonsDataFrame(sr, data.frame(row.names=c('1','2','3','4'), PIDS=1:4))

# Intersect points with polygons and add polygon IDS to pts@data.
```

```
  pts.poly <- point.in.poly(meuse, srdf)
    head(pts.poly@data)

# Point counts for each polygon
tapply(pts.poly@data$lead, pts.poly@data$PIDS, FUN=length)
```

---

pp.subsample *Point process random subsample*

Description

Generates random subsample based on density estimate of observations

Usage

```
pp.subsample(x, n, window = "hull", sigma = "Scott", wts = NULL,
  gradient = 1, edge = FALSE)
```

Arguments

x	An sp class SpatialPointsDataFrame or SpatialPoints object
n	Number of random samples to generate
window	Type of window (hull or extent)
sigma	Bandwidth selection method for KDE, default is 'Scott'. Options are 'Scott', 'Stoyan', 'Diggle', 'likelihood', and 'geometry'
wts	Optional vector of weights corresponding to point pattern
gradient	A scaling factor applied to the sigma parameter used to adjust the gradient decent of the density estimate. The default is 1, for no adjustment (downweight < 1 | upweight > 1)
edge	Apply Diggle edge correction (TRUE/FALSE)

Value

sp class SpatialPointsDataFrame containing random subsamples

Note

The window type creates a convex hull by default or, optionally, uses the maximum extent (envelope).

Available bandwidth selection methods are: Scott (Scott 1992), Scott's Rule for Bandwidth Selection (1st order) Diggle (Berman & Diggle 1989), Minimise the mean-square error via cross validation (2nd order) likelihood (Loader 1999), Maximum likelihood cross validation (2nd order) geometry, Bandwidth is based on simple window geometry (1st order) Stoyan (Stoyan & Stoyan 1995), Based on pair-correlation function (strong 2nd order)

Note; resulting bandwidth can vary widly by method. the 'diggle' method is intended for selecting bandwidth representing 2nd order spatial variation whereas the 'scott' method will represent 1st order trend. the 'geometry' approach will also represent 1st order trend. for large datasets, caution should be used with the 2nd order 'likelihood' approach, as it is slow and computationaly expensive. finally, the 'stoyan' method will produce very strong 2nd order results. '

Depends: sp, spatstat

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Berman, M. and Diggle, P. (1989) Estimating weighted integrals of the second-order intensity of a spatial point process. Journal of the Royal Statistical Society, series B 51, 81-92. Berman, M. and Diggle, P. (1989) Estimating weighted integrals of the second-order intensity of a spatial point process. Journal of the Royal Statistical Society, series B 51, 81-92.

Fithian, W & T. Hastie (2013) Finite-sample equivalence in statistical models for presence-only data. Annals of Applied Statistics 7(4): 1917-1939

Hengl, T., H. Sierdsema, A. Radovic, and A. Dilo (2009) Spatial prediction of species distributions from occurrence-only records: combining point pattern analysis, ENFA and regression-kriging. Ecological Modelling, 220(24):3499-3511

Loader, C. (1999) Local Regression and Likelihood. Springer, New York.

Scott, D.W. (1992) Multivariate Density Estimation. Theory, Practice and Visualization. New York, Wiley.

Stoyan, D. and Stoyan, H. (1995) Fractals, random shapes and point fields: methods of geometrical statistics. John Wiley and Sons.

Warton, D.i., and L.C. Shepherd (2010) Poisson Point Process Models Solve the Pseudo-Absence Problem for Presence-only Data in Ecology. The Annals of Applied Statistics, 4(3):1383-1402

Examples

```
require(spatstat)
require(sp)
data(bei)
  trees <- as(bei, 'SpatialPoints')
    n=round(length(trees) * 0.10, digits=0)
      trees.wrs <- pp.subsample(trees, n=n, window='hull')
        plot(trees, pch=19, col='black')
          plot(trees.wrs, pch=19, col='red', add=TRUE)
            box()
             title('10% subsample')
          legend('bottomright', legend=c('Original sample', 'Subsample'),
                 col=c('black','red'),pch=c(19,19))
```

print.loess.boot *Print Loess bootstrap model*

Description

print method for class "loess.boot"

Usage

```
## S3 method for class 'loess.boot'
print(x, ...)
```

Arguments

x	Object of class loess.boot
...	Ignored

pseudo.absence *Pseudo-absence random samples*

Description

Generates pseudo-absence samples based on density estimate of known locations

Usage

```
pseudo.absence(x, n, window = "hull", Mask = NULL, s = NULL,
  sigma = "Scott", wts = NULL, KDE = FALSE, gradient = 1, p = NULL,
  edge = FALSE)
```

Arguments

x	An sp class SpatialPointsDataFrame or SpatialPoints object
n	Number of random samples to generate
window	Type of window (hull OR extent), overridden if mask provided
Mask	Optional rasterLayer class mask raster. The resolution of the density estimate will match mask.
s	Optional resolution passed to window argument. Caution should be used due to long processing times associated with high resolution. In contrast, coarse resolution can exclude known points.
sigma	Bandwidth selection method for KDE, default is 'Scott'. Options are 'Scott', 'Stoyan', 'Diggle', 'likelihood', and 'geometry'
wts	Optional vector of weights corresponding to point pattern

KDE	save KDE raster (TRUE/FALSE)
gradient	A scaling factor applied to the sigma parameter used to adjust the gradient decent of the density estimate. The default is 1, for no adjustment (downweight < 1 | upweight > 1)
p	Minimum value for probability distribution (must be > 0)
edge	Apply Diggle edge correction (TRUE/FALSE)

Value

A list class object with the following components: @return sample SpatialPointsDataFrame containing random samples @return kde sp RasterLayer class of KDE estimates (IF KDE = TRUE) @return sigma Selected bandwidth of KDE

Note

The window type creates a convex hull by default or, optionally, uses the maximum extent (envelope). if a mask is provided the kde will represent areas defined by the mask. this defines the area that pseudo absence data will be generated.'

Available bandwidth selection methods are: Scott (Scott 1992), Scott's Rule for Bandwidth Selection (1st order) Diggle (Berman & Diggle 1989), Minimise the mean-square error via cross validation (2nd order) likelihood (Loader 1999), Maximum likelihood cross validation (2nd order) geometry, Bandwidth is based on simple window geometry (1st order) Stoyan (Stoyan & Stoyan 1995), Based on pair-correlation function (strong 2nd order)

Note; resulting bandwidth can vary widely by method. the 'diggle' method is intended for selecting bandwidth representing 2nd order spatial variation whereas the 'scott' method will represent 1st order trend. the 'geometry' approach will also represent 1st order trend. for large datasets, caution should be used with the 2nd order 'likelihood' approach, as it is slow and computationally expensive. finally, the 'stoyan' method will produce very strong 2nd order results.

Depends: sp, spatstat, raster

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Berman, M. and Diggle, P. (1989) Estimating weighted integrals of the second-order intensity of a spatial point process. Journal of the Royal Statistical Society, series B 51, 81-92.

Fithian, W & T. Hastie (2013) Finite-sample equivalence in statistical models for presence-only data. Annals of Applied Statistics 7(4): 1917-1939

Hengl, T., H. Sierdsema, A. Radovic, and A. Dilo (2009) Spatial prediction of species distributions from occurrence-only records: combining point pattern analysis, ENFA and regression-kriging. Ecological Modelling, 220(24):3499-3511

Loader, C. (1999) Local Regression and Likelihood. Springer, New York.

Scott, D.W. (1992) Multivariate Density Estimation. Theory, Practice and Visualization. New York, Wiley.

Stoyan, D. and Stoyan, H. (1995) Fractals, random shapes and point fields: methods of geometrical statistics. John Wiley and Sons.

Warton, D.i., and L.C. Shepherd (2010) Poisson Point Process Models Solve the Pseudo-Absence Problem for Presence-only Data in Ecology. The Annals of Applied Statistics, 4(3):1383-1402

Examples

```
library(sp)
data(meuse)
  coordinates(meuse) = ~x+y

pa <- pseudo.absence(meuse, n=100, window='hull', KDE=TRUE, sigma='Diggle', s=50)
  col.br <- colorRampPalette(c('blue','yellow'))
    plot(pa$kde, col=col.br(10))
      plot(meuse, pch=20, cex=1, add=TRUE)
        plot(pa$sample, col='red', pch=20, cex=1, add=TRUE)
          legend('top', legend=c('Presence', 'Pseudo-absence'),
                 pch=c(20,20),col=c('black','red'))

# With clustered data
library(sp)
library(spatstat)
data(bei)
  trees <- as(bei, 'SpatialPoints')
    trees <- SpatialPointsDataFrame(coordinates(trees),
                         data.frame(ID=1:length(trees)))
      trees.abs <- pseudo.absence(trees, n=100, window='extent', KDE=TRUE)

col.br <- colorRampPalette(c('blue','yellow'))
  plot(trees.abs$kde, col=col.br(10))
   plot(trees, pch=20, cex=0.50, add=TRUE)
      plot(trees.abs$sample, col='red', pch=20, cex=1, add=TRUE)
        legend('top', legend=c('Presence', 'Pseudo-absence'),
               pch=c(20,20),col=c('black','red'))
```

---

pu *Biodiversity Planning Units*

Description

Subset of biodiversity planning units for Haiti ecoregional spatial reserve plan

Format

A sp SpatialPolygonsDataFrame with 5919 rows and 46 variables:

UNIT_ID Unique planning unit ID

DR_Dr_A Biodiversity target

DR_Dr_L Biodiversity target
Ht_Dr_A Biodiversity target
Ht_Dr_L Biodiversity target
DR_Ms_A Biodiversity target
DR_Ms_L Biodiversity target
Ht_Ms_L Biodiversity target
DR_LM_M Biodiversity target
H_LM_M_L Biodiversity target
H_LM_R_L Biodiversity target
DR_LM_R_L Biodiversity target
DR_Rn_L Biodiversity target
DR_LM_R_S Biodiversity target
DR_Rn_S Biodiversity target
DR_Ms_S Biodiversity target
Ht_Ms_A Biodiversity target
DR_Ms_E Biodiversity target
DR_Ms_I Biodiversity target
DR_Rn_E Biodiversity target
DR_Rn_I Biodiversity target
H_LM_R_E Biodiversity target
Ht_Ms_E Biodiversity target
Ht_Rn_E Biodiversity target
DR_Rn_A Biodiversity target
Ht_Rn_A Biodiversity target
Ht_Rn_I Biodiversity target
Ht_Dr_E Biodiversity target
Ht_Ms_S Biodiversity target
Ht_Dr_S Biodiversity target
Ht_Rn_L Biodiversity target
Ht_Th_A Biodiversity target
Ht_Th_L Biodiversity target
Ht_Th_S Biodiversity target
Ht_Dr_U Biodiversity target
Ht_Dr_I Biodiversity target
Ht_Ms_I Biodiversity target
H_LM_M_A Biodiversity target
H_LM_M_E Biodiversity target

H_LM_R_A Biodiversity target

H_LM_M_S Biodiversity target

H_LM_R_I Biodiversity target

H_LM_R_S Biodiversity target

Ht_Rn_S Biodiversity target

Ht_Ms_U Biodiversity target

Ht_Rn_U Biodiversity target

Source

http://maps.tnc.org/gis_data.html

References

Evans, J.S., S.R. Schill, G.T. Raber (2015) A Systematic Framework for Spatial Conservation Planning and Ecological Priority Design in St. Lucia, Eastern Caribbean. Chapter 26 in Central American Biodiversity : Conservation, Ecology and a Sustainable Future. F. Huettman (eds). Springer, NY.

`raster.entropy` *Raster Entropy*

Description

Calculates entropy on integer raster (i.e., 8 bit 0-255)

Usage

```
raster.entropy(x, d = 5, filename = FALSE, ...)
```

Arguments

`x`	object of class raster (requires integer raster)
`d`	Size of matrix (window)
`filename`	Raster file written to disk
`...`	Optional arguments passed to writeRaster or dataType

Value

raster class object or specified format raster written to disk

Note

Entropy calculated as: H = -sum(Pi*ln(Pi)) where; Pi, Proportion of one value to total values Pi=n(p)/m and m, Number of unique values Expected range: 0 to log(m) H=0 if window contains the same value in all cells. H increases with the number of different values in the window.

Maximum entropy is reached when all values are different, same as log(m) max.ent <- function(x) log(length(unique(x)))

Depends: raster

References

Fuchs M., Hoffmann R., Schwonke F. (2008) Change Detection with GRASS GIS - Comparison of images taken by different sensor. On line at: http://geoinformatics.fsv.cvut.cz/gwiki/Change_Detection_with_GRASS_GIS_-_Comparison_of_images_taken_by_different_sensors

Examples

```
require(raster)
  r <- raster(ncols=100, nrows=100)
    r[] <- round(runif(ncell(r), 1,8), digits=0)

rEnt <- raster.entropy(r, d=5)
  opar <- par
    par(mfcol=c(2,1))
      plot(r)
        plot(rEnt)
  par(opar)
```

---

raster.vol *Raster Percent Volume*

Description

Calculates a percent volume on a raster or based on a systematic sample

Usage

```
raster.vol(x, p = 0.95, sample = FALSE, spct = 0.05)
```

Arguments

x	raster class object
p	percent raster-value volume
sample	base volume on systematic point sample (TRUE/FALSE)
spct	sample percent, if sample (TRUE)

Value

if sample (FALSE) binary raster object with 1 representing designated percent volume

if sample (TRUE) n sp SpatialPointsDataFrame object with points that represent the percent volume of the sub-sample

Note

Since this model needs to operate on all of the raster values, it is not memory safe

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
require(raster)
  r <- raster(ncols=100, nrows=100)
    r[] <- runif(ncell(r), 0, 1)
    r <- focal(r, w=focalWeight(r, 6, "Gauss"))
    r[sample(1000, 1:ncell(r))] <- NA

  # full raster percent volume
  p30 <- raster.vol(r, p=0.30)
  p50 <- raster.vol(r, p=0.50)
  p80 <- raster.vol(r, p=0.80)
    par(mfrow=c(2,2))
    plot(r, col=cm.colors(10), main="original raster")
    plot(p30, breaks=c(0,0.1,1), col=c("cyan","red"), legend=FALSE,
      main="30% volume")
    plot(p50, breaks=c(0,0.1,1), col=c("cyan","red"), legend=FALSE,
      main="50% volume")
    plot(p80, breaks=c(0,0.1,1), col=c("cyan","red"), legend=FALSE,
      main="80% volume")

  # point sample percent volume
  #  p30 <- raster.vol(r, p = 0.30, sample = TRUE, spct = 0.20)
  #    plot(r, main="30% volume point sample")
  #    plot(p30, pch=20, cex=0.70, add=TRUE)
```

---

sample.line *Systematic or random point sample of line(s)*

Description

Creates a systematic or random point sample of an sp SpatialLinesDataFrame object based on distance spacing, fixed size or proportional size

Usage

```
sample.line(x, d = 100, p = NULL, n = NULL, type = "regular",
  longlat = FALSE, min.samp = 1, ...)
```

Arguments

x	sp class SpatialLinesDataFrame object
d	Sample distance. For regular sample.
p	Proportional sample size (length * p), expected value is 0-1. For regular or random.
n	Fixed sample size. For regular or random
type	Defines sample type. Options are "regular" or "random". A regular sample results in a systematic, evenly spaced sample.
longlat	TRUE/FALSE is data in geographic units, if TRUE distance is in kilometres
min.samp	Minimal number of sample points for a given line (default is 1 point)
...	Additional argument passed to spsample

Value

sp SpatialPointsDataFrame object.

Note

The sdist argument will produce an evenly spaced sample, whereas n produces a fixed sized sample. The p (proportional) argument calculates the percent of the line-length.

The LID column in the @data slot corresponds to the row.names of the SpatialLinesDataFrame object.

Depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
 require(sp)
 sp.lines <- SpatialLines(list(Lines(list(Line(cbind(c(1,2,3),c(3,2,2)))), ID="1"),
                          Lines(list(Line(cbind(c(1,2,3),c(1,1.5,1)))), ID="2")))
 sp.lines <- SpatialLinesDataFrame( sp.lines, data.frame(ID=1:2, row.names=c(1,2)) )

par(mfrow=c(2,2))
 # Create systematic sample at 20 km spacing
 reg.sample <- sample.line(sp.lines, d = 20, type = "regular", longlat = TRUE)
   plot(sp.lines)
     plot(reg.sample, pch = 20, add = TRUE)
  box()
  title("systematic d = 20")
```

```
# Create fixed size (n = 20) systematic sample
reg.sample <- sample.line(sp.lines, n = 20, type = "regular", longlat = TRUE)
  plot(sp.lines)
    plot(reg.sample, pch = 20, add = TRUE)
    box()
 title("systematic n = 20")

# Create fixed size (n = 20) random sample
rand.sample <- sample.line(sp.lines, n = 20, type = "random", longlat = TRUE)
  plot(sp.lines)
    plot(rand.sample, pch = 20, add = TRUE)
    box()
    title("rand n = 20")

# Create proportional (p = 0.10) random sample
rand.sample <- sample.line(sp.lines, p = 0.10, type = "random", longlat = TRUE)
  plot(sp.lines)
    plot(rand.sample, pch = 20, add = TRUE)
    box()
 title("rand p = 0.10")
```

`sample.poly` *Sample Polygons*

Description

Creates an equal sample of n for each polygon in an sp Polygon class object

Usage

```
sample.poly(x, n = 10, type = "random", ...)
```

Arguments

`x`	sp class SpatialPolygons or SpatialPolygonsDataFrame object
`n`	Number of random samples
`type`	Type of sample with options for: "random", "regular", "stratified", "nonaligned", "hexagonal", "clustered", "Fibonacci". See "spsample" for details.
`...`	Additional arguments passed to spsample

Value

sp SpatialPointsDataFrame object

Note

Depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
library(raster)
library(sp)
  p <- raster(nrow=10, ncol=10)
  p[] <- runif(ncell(p)) * 10
  p <- rasterToPolygons(p, fun=function(x){x > 9})
  s <- sample.poly(p, n = 5, type = "random")
    plot(p)
      plot(s, pch = 20, add = TRUE)
      box()
      title("Random sample (n=5) for each polygon")
```

---

separability *separability*

Description

Calculates variety of univariate or multivariate separability metrics for two class samples

Usage

```
separability(x, y, plot = FALSE, cols = c("red", "blue"),
  clabs = c("Class1", "Class2"), ...)
```

Arguments

x	X vector
y	Y vector
plot	plot separability (TRUE/FALSE)
cols	colours for plot (must be equal to number of classes)
clabs	labels for two classes
...	additional arguments passes to plot

Value

A data.frame with the following separability metrics:

B Bhattacharryya distance statistic

JM Jeffries-Matusita distance statistic

M M-Statistic

D Divergence index

TD Transformed Divergence index

Note

M-Statistic (Kaufman & Remer 1994) - This is a measure of the difference of the distributional peaks. A large M-statistic indicates good separation between the two classes as within-class variance is minimized and between-class variance maximized (M <1 poor, M >1 good).

Bhattacharyya distance (Bhattacharyya 1943; Harold 2003) - Measures the similarity of two discrete or continuous probability distributions.

Jeffries-Matusita distance (Bruzzone et al., 2005; Swain et al., 1971) - The J-M distance is a function of separability that directly relates to the probability of how good a resultant classification will be. The J-M distance is asymptotic to v2, where values of v2 suggest complete separability

Divergence and transformed Divergence (Du et al., 2004) - Maximum likelihood approach. Transformed divergence gives an exponentially decreasing weight to increasing distances between the classes.

Author(s)

Jeffrey S. Evans <jeffrey_evans<at>tnc.org>

References

Anderson, M. J., & Clements, A. (2000) Resolving environmental disputes: a statistical method for choosing among competing cluster models. Ecological Applications 10(5):1341-1355

Bhattacharyya, A. (1943) On a measure of divergence between two statistical populations defined by their probability distributions'. Bulletin of the Calcutta Mathematical Society 35:99-109

Bruzzone, L., F. Roli, S.B. Serpico (1995) An extenstion to multiclass cases of the Jefferys-Matusita distance. IEEE Transactions on Pattern Analysis and Machine Intelligence 33:1318-1321

Du, H., C.I. Chang, H. Ren, F.M. D'Amico, J. O. Jensen, J., (2004) New Hyperspectral Discrimination Measure for Spectral Characterization. Optical Engineering 43(8):1777-1786.

Kailath, T., (1967) The Divergence and Bhattacharyya measures in signal selection. IEEE Transactions on Communication Theory 15:52-60

Kaufman Y., and L. Remer (1994) Detection of forests using mid-IR reflectance: An application for aerosol studies. IEEE T. Geosci.Remote. 32(3):672-683.

Examples

```
norm1 <- dnorm(seq(-20,20,length=5000),mean=0,sd=1)
norm2 <- dnorm(seq(-20,20,length=5000),mean=0.2,sd=2)
  separability(norm1, norm2)

s1 <- c (1362,1411,1457,1735,1621,1621,1791,1863,1863,1838)
s2 <- c (1362,1411,1457,10030,1621,1621,1791,1863,1863,1838)
  separability(s1, s2, plot=TRUE)
```

shannons *Shannon's Diversity (Entropy) Index*

Description

Calculates Shannon's Diversity Index and Shannon's Evenness Index

Usage

```
shannons(x, counts = TRUE, margin = "row")
```

Arguments

x	data.frame object containing species counts
counts	Are data counts (TRUE) or relative proportions (FALSE)
margin	Calculate diversity for rows or columns. c("row", "col")

Value

data.frame with columns "H" (Shannon's diversity) and "evenness" (Shannon's evenness where H / max(sum(x)))

Note

The sdist argument will produce an evenly spaced sample, whereas n produces a fixed sized sample. The p (proportional) argument calculates the percent of the line-length.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Shannon, C. E. and W. Weaver (1948) A mathematical theory of communication. The Bell System Technical Journal, 27:379-423. Simpson, E. H. (1949) Measurement of diversity. Nature 163:688 Roth, D. S., I. Perfecto, and B. Rathcke (1994) The effects of management systems on ground-foraging ant diversity in Costa Rica. Ecological Applications 4(3):423-436.

Examples

```
# Using Costa Rican ant diversity data from Roth et al. (1994)
data(ants)

# Calculate diversity for each covertype ("col")
shannons(ants[,2:ncol(ants)], counts = FALSE, margin = "col")

# Calculate diversity for each species ("row")
ant.div <- shannons(ants[,2:ncol(ants)], counts = FALSE, margin = "row")
```

```
names(ant.div) <- ants[,1]
ant.div
```

`similarity` *Ecological similarity*

Description

Uses row imputation to identify "k" ecological similar observations

Usage

```
similarity(x, k = 4, method = "mahalanobis", frequency = TRUE,
  scale = TRUE, ID = NULL)
```

Arguments

`x`	data.frame containing ecological measures
`k`	Number of k nearest neighbors (kNN)
`method`	Method to compute multivariate distances c("mahalanobis", "raw", "euclidean", "ica")
`frequency`	Calculate frequency of each reference row (TRUE/FALSE)
`scale`	Scale multivariate distances to standard range (TRUE/FALSE)
`ID`	Unique ID vector to use as reference ID's (rownames). Must be unique and same length as number of rows in x

Value

data.frame with k similar targets and associated distances. If frequency = TRUE the freq column represents the number of times a row (ID) was selected as a neighbor.

Note

This function uses row-based imputation to identify k similar neighbors for each observation. Has been used to identify offsets based on ecological similarity.

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Evans, J.S., S.R. Schill, G.T. Raber (2015) A Systematic Framework for Spatial Conservation Planning and Ecological Priority Design in St. Lucia, Eastern Caribbean. Chapter 26 in Central American Biodiversity : Conservation, Ecology and a Sustainable Future. F. Huettman (eds). Springer, NY.

Examples

```
 data(pu)
 kNN <- similarity(pu@data[2:ncol(pu)], k = 4, frequency = FALSE, ID = pu@data$UNIT_ID)

## Not run:
 kNN <- similarity(pu@data[2:ncol(pu)], k = 4, frequency = TRUE, ID = pu@data$UNIT_ID)
 p <- kNN$freq
 clr <- c("#3288BD", "#99D594", "#E6F598", "#FEE08B",
          "#FC8D59", "#D53E4F")
 p <- ifelse(p <= 0, clr[1],
        ifelse(p > 0 & p < 10, clr[2],
          ifelse(p >= 10 & p < 20, clr[3],
         ifelse(p >= 20 & p < 50, clr[4],
           ifelse(p >= 50 & p < 100, clr[5],
             ifelse(p >= 100, clr[6], NA))))))
 plot(pu, col=p, border=NA)
   legend("topleft", legend=c("None","<10","10-20",
          "20-50","50-100",">100"),
          fill=clr, cex=0.6, bty="n")
   box()

## End(Not run)
```

`sp.na.omit` *sp na.omit*

Description

Removes row or column NA's in sp object

Usage

```
sp.na.omit(x, margin = 1)
```

Arguments

`x`	Object of class SpatialPointsDataFrame OR SpatialPolygonsDataFrame
`margin`	Margin (1,2) of data.frame 1 for rows or 2 for columns

Note

Depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans<at>tnc.org>

Examples

```
require(sp)
  data(meuse)
    coordinates(meuse) <- ~x+y

   # Display rows with NA
   meuse@data[!complete.cases(meuse@data),]

   # Remove NA's in rows (and associated points)
   meuse2 <- sp.na.omit(meuse)
     dim(meuse)
       dim(meuse2)

   # Plot deleted points in red
   plot(meuse, col='red', pch=20)
     plot(meuse2, col='black', pch=20, add=TRUE)

   # Remove columns with NA's
   meuse2 <- sp.na.omit(meuse, margin=2)
     head(meuse@data)
       head(meuse2@data)
```

---

stratified.random *Stratified random sample*

Description

Creates a stratified random sample of an sp class object

Usage

```
stratified.random(x, strata, n = 10, reps = 1, replace = TRUE)
```

Arguments

x	sp class SpatialDataFrame object (point, polygon, line, pixel)
strata	Column in @data slot with stratification factor
n	Number of random samples
reps	Number of replicates per strata
replace	Sampling with replacement (TRUE|FALSE)

Value

sp SpatialDataFrame object (same as input feature) containing random samples

Note

If replace=FALSE features are removed from consideration in subsequent replicates. Conversely, if replace=TRUE, a feature can be selected multiple times across replicates. Not applicable if rep=1.

Depends: sp

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Hudak, A.T., N.L. Crookston, J.S. Evans, M.J. Falkowski, A.M.S. Smith, P. Gessler and P. Morgan. (2006) Regression modelling and mapping of coniferous forest basal area and tree density from discrete-return lidar and multispectral satellite data. Canadian Journal of Remote Sensing 32: 126-138.

Examples

```
require(sp)
  data(meuse)
    coordinates(meuse) <- ~x+y

# Create stratified variable using quartile breaks
x1 <- cut(meuse@data[,'cadmium'], summary(meuse@data[,'cadmium'])[-4], include.lowest=TRUE)
  levels(x1) <- seq(1,nlevels(x1),1)
x2 <- cut(meuse@data[,'lead'], summary(meuse@data[,'lead'])[-4], include.lowest=TRUE)
  levels(x2) <- seq(1,nlevels(x2),1)
meuse@data <- cbind(meuse@data, STRAT=paste(x1, x2, sep='.') )

# 2 replicates and replacement
ssample <- stratified.random(meuse, strata='STRAT', n=2, reps=2)

# 2 replicates and no replacement
ssample.nr <- stratified.random(meuse, strata='STRAT', n=2, reps=2, replace=FALSE)

# n=1 and reps=10 for sequential numbering of samples
ssample.ct <- stratified.random(meuse, strata='STRAT', n=1, reps=10, replace=TRUE)

# Counts for each full strata (note; 2 strata have only 1 observsation)
tapply(meuse@data$STRAT, meuse@data$STRAT, length)

# Counts for each sampled strata, with replacement
tapply(ssample@data$STRAT, ssample@data$STRAT, length)

# Counts for each sampled strata, without replacement
tapply(ssample.nr@data$STRAT, ssample.nr@data$STRAT, length)

# Counts for each sampled strata, without replacement
tapply(ssample.ct@data$STRAT, ssample.ct@data$STRAT, length)

# Plot random samples colored by replacement
```

```
ssample@data$REP <- factor(ssample@data$REP)
  spplot(ssample, 'REP', col.regions=c('red','blue'))
```

---

summary.loess.boot *Summarizing Loess bootstrap models*

Description

Summary method for class "loess.boot".

Usage

```
## S3 method for class 'loess.boot'
summary(object, ...)
```

Arguments

object	Object of class loess.boot
...	Ignored

---

tpi *Topographic Position Index (tpi)*

Description

Calculates topographic position using mean deviations

Usage

```
tpi(x, scale = 3, win = "rectangle", normalize = FALSE)
```

Arguments

x	raster class object
scale	scale (window size)
win	window type. Options are "rectangle" and "circle"
normalize	Apply deviation correction that normalizes to local surface roughness

Value

raster class object of tpi

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

De Reu, J., J. Bourgeois, M. Bats, A. Zwertvaegher, V. Gelorini, et al., (2014) Application of the topographic position index to heterogeneous landscapes. Geomorphology, 186:39-49.

Examples

```
 library(raster)
 r <- raster(nrows=500, ncols=500, xmn=571823, xmx=616763,
             ymn=4423540, ymx=4453690)
 proj4string(r) <- crs("+proj=utm +zone=12 +datum=NAD83 +units=m +no_defs")
 r[] <- runif(ncell(r), 1000, 2500)
 r <- focal(r, focalWeight(r, 150, "Gauss") )

# calculate tpi and plot
  tpi9 <- tpi(r, scale=9)
    par(mfrow=c(1,2))
      plot(r, main="original raster")
      plot(tpi9, main="tpi 9x9")
```

`trend.line` *trend.line*

Description

Calculated specified trend line of x,y

Usage

```
trend.line(x, y, type = "linear", plot = TRUE, ...)
```

Arguments

`x`	Vector of x
`y`	Vector of y
`type`	Trend line types are: 'linear', 'exponential', 'logarithmic', 'polynomial'
`plot`	plot results (TRUE/FALSE)
`...`	Additional arguments passed to plot

Value

A list class object with the following components: for type = 'linear' x is slope and y is intercept for type = 'exponential', 'logarithmic', or 'polynomial' x is original x variable and y is vector of fit regression line

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
x <- 1:10
y <- jitter(x^2)
  opar <- par
  par(mfcol=c(2,2))
    trend.line(x,y,type='linear',plot=TRUE,pch=20,main='Linear')
    trend.line(x,y,type='exponential',plot=TRUE,pch=20,main='Exponential')
    trend.line(x,y,type='logarithmic',plot=TRUE,pch=20,main='Logarithmic')
    trend.line(x,y,type='polynomial',plot=TRUE,pch=20,main='Polynomial')
 par <- opar
```

---

tri *Terrain Ruggedness Index*

Description

Implementation of the Riley et al (1999) Terrain Ruggedness Index

Usage

```
tri(r, s = 3, exact = TRUE, file.name = NULL, ...)
```

Arguments

r	raster class object
s	Scale of window. Must be odd number, can represent 2 dimensions (eg., s=c(3,5) would represent a 3 x 5 window)
exact	Calculate (TRUE/FALSE) the exact TRI or an algebraic approximation.
file.name	Name of output raster (optional)
...	Additional arguments passed to writeRaster

Value

raster class object or raster written to disk

Note

The algebraic approximation is considerably faster. However, because inclusion of the center cell, the larger the scale the larger the divergence of the minimum value

Recommended ranges for classifying Topographic Ruggedness Index

0-80 (1) level terrain surface.

81-116 (2) nearly level surface.

117-161 (3) slightly rugged surface.

162-239 (4) intermediately rugged surface.

240-497 (5) oderately rugged surface.

498-958 (6) highly rugged surface.

>959 (7) extremely rugged surface.

Depends: raster

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

References

Riley, S.J., S.D. DeGloria and R. Elliot (1999) A terrain ruggedness index that quantifies topographic heterogeneity, Intermountain Journal of Sciences 5(1-4):23-27.

Examples

```
library(raster)
r <- raster(nrows=180, ncols=360, xmn=571823.6, xmx=616763.6, ymn=4423540,
            ymx=4453690, resolution=270, crs = CRS("+proj=utm +zone=12 +datum=NAD83
            +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0"))
  r[] <- runif(ncell(r), 1000, 5000)
  r <- focal(r, w=matrix(1/121,nrow=11,ncol=11))

 ( tri.ext <- tri(r) )
 ( tri.app <- tri(r, exact = FALSE) )
 plot(stack(tri.ext, tri.app))
```

---

wt.centroid *Weighted centroid*

Description

Creates centroid of [x,y] coordinates based on a weights field

Usage

```
wt.centroid(x, p, sp = TRUE)
```

Arguments

x	sp SpatialPointsDataFrame class object
p	Weights column in x@data slot
sp	Output sp SpatailPoints class object (TRUE | FALSE)

Value

A vector or an sp class SpatialPoints object of the weighted coordinate centroid

Note

THe weighted centroid is calculated as: [Xw]=[X]*[p], [Yw]=[Y]*[p], [sXw]=SUM[Xw], [sYw]=SUM[Yw], [sP]=SUM[p] wX=[sXw]/[sP], wY=[sYw]/[sP] where; X=X COORDINATE(S), Y=Y COORDINATE(S), p=WEIGHT

Depends: sp

Examples

```
require(sp)
 data(meuse)
 coordinates(meuse) = ~x+y
 wt.copper <- wt.centroid(meuse, 'copper', sp=TRUE)
   wt.zinc <- wt.centroid(meuse, 'zinc', sp=TRUE)
     plot(meuse, pch=20, cex=0.75, main='Weighted centroid(s)')
       points(wt.copper, pch=19, col='red', cex=1.5)
         points(wt.zinc, pch=19, col='blue', cex=1.5)
        box()
legend('topleft', legend=c('all','copper', 'zinc'),
       pch=c(20,19,19),col=c('black','red','blue'))
```

`zonal.stats` *zonal.stats*

Description

Polygon zonal statistics of a raster

Usage

```
zonal.stats(x, y, stat, trace = TRUE, plot = TRUE)
```

Arguments

`x`	Polygon object of class SpatialPolygonsDataFrame
`y`	Raster object of class raster
`stat`	Statistic or function
`trace`	Should progress counter be displayed
`plot`	Should subset polygons/rasters be plotted (TRUE/FALSE)

Value

Vector, length of nrow(x), of function results

Note

This function iterates through a polygon object, masks the raster to each subset polygon and then coerces the subset raster to a vector object. The resulting vector is then passed to the specified statistic/function. This is much slower than zonal functions available in GIS software but has the notable advantage of being able to accept any custom function, passed to the 'stat' argument, appropriate for a vector object (see example).

Depends: sp, raster

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org>

Examples

```
# skewness function
skew <- function(x, na.rm=FALSE) {
   if (na.rm)
       x <- x[!is.na(x)]
  sum( (x - mean(x)) ^ 3) / ( length(x) * sd(x) ^ 3 )
  }

# percent x >= p function
pct <- function(x, p=0.30) {
  if ( length(x[x >= p]) < 1 )  return(0)
    if ( length(x[x >= p]) == length(x) ) return(1)
     else return( length(x[x >= p]) / length(x) )
}

# create some example data
library(raster)
library(sp)
p <- raster(nrow=10, ncol=10)
  p[] <- runif(ncell(p)) * 10
    p <- rasterToPolygons(p, fun=function(x){x > 9})
      r <- raster(nrow=100, ncol=100)
        r[] <- runif(ncell(r))
plot(r)
  plot(p, add=TRUE, lwd=4)

# run zonal statistics using skew and pct functions
z.skew <- zonal.stats(x=p, y=r, stat=skew, trace=TRUE, plot=TRUE)
z.pct <- zonal.stats(x=p, y=r, stat=pct, trace=TRUE, plot=TRUE)
  ( z <- data.frame(ID=as.numeric(as.character(row.names(p@data))),
                    SKEW=z.skew, PCT=z.pct) )
```

Index