Package 'ads'

October 10, 2025

Type Package

Title Spatial Point Patterns Analysis

Version 1.5-11

Date 2025-10-10

Maintainer Dominique Lamonica <dominique.lamonica@ird.fr>

URL https://forge.ird.fr/amap/ads

BugReports https://forge.ird.fr/amap/ads/-/issues

Imports ade4, spatstat.geom

Description Perform first- and second-order multi-scale analyses derived from Ripley K-function (Ripley B. D. (1977) <doi:10.1111/j.2517-6161.1977.tb01615.x>), for univariate, multivariate and marked mapped data in rectangular, circular or irregular shaped sampling windows, with tests of
statistical significance based on Monte Carlo simulations.

Depends R (>= 3.5.0)

License GPL-2

NeedsCompilation yes

Repository CRAN

RoxygenNote 7.3.2

Language en-GB

Author Raphael Pelissier [aut],
Francois Goreaud [aut],
Philippe Verley [ctb],
Dominique Lamonica [cre]

Date/Publication 2025-10-10 11:40:02 UTC

Contents

`Allogny` *Spatial pattern of oaks suffering from frost shake in Allogny, France.*

Description

Spatial pattern of sound and split oaks (*Quercus petraea*) suffering from frost shake in a 2.35-ha plot in Allogny, France.

Usage

```
data(Allogny)
```

Format

A list with 4 components:
`$rect` is a vector of coordinates $(xmin, ymin, xmax, ymax)$ of the origin and the opposite corner of a 125 by 188 m square plot.
`$trees` is a list of tree coordinates (x, y).
`$status` is a factor with 2 levels ("*splited*","*sound*").

Source

Grandjean, G., Jabiol, B., Bruchiamacchie, M. and Roustan, F. 1990. *Recherche de correlations entre les parametres edaphiques, et plus specialement texture, hydromorphie et drainage interne, et la reponse individuelle des chenes sessiles et pedoncules ? la gelivure.* Rapport de recherche ENITEF, Nogent sur Vernisson, France.

References

Goreaud, F. & Pelissier, R. 2003. Avoiding misinterpretation of biotic interactions with the intertype *K12*-function: population independence vs. random labelling hypotheses. *Journal of Vegetation Science*, 14: 681-692.

Examples

```
data(Allogny)
allo.spp <- spp(Allogny$trees, mark=Allogny$status, window=Allogny$rect)
plot(allo.spp)
```

---

area.swin *Area of a sampling window*

Description

Function `area.swin` computes the area of a sampling window.

Usage

```
area.swin(w)
```

Arguments

`w`	an object of class `"swin"` defining the sampling window.

Details

For `"simple"` sampling windows, returns simply the area of the rectangle or circle delineating the study region.
For `"complex"` sampling windows, returns the area of the initial rectangle or circle, minus the total area of the triangles to remove (see `swin`).

Value

The area of the sampling window.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

swin.

Examples

```
## Not run: rectangle of size [0,110] x [0,90]
wr<-swin(c(0,0,110,90))
area.swin(wr)

## Not run: circle with radius 50 centred on (55,45)
wc<-swin(c(55,45,50))
area.swin(wc)

## Not run: polygon (diamond shape)
t1 <- c(0,0,55,0,0,45)
t2 <- c(55,0,110,0,110,45)
t3 <- c(0,45,0,90,55,90)
t4 <- c(55,90,110,90,110,45)
wp <- swin(wr, rbind(t1,t2,t3,t4))
area.swin(wp)
```

BPoirier *Tree spatial pattern in Beau Poirier plot, Haye forest, France*

Description

Spatial pattern of 162 beeches, 72 oaks and 3 hornbeams in a 1-ha 140 yr-old temperate forest plot in Haye, France.

Usage

```
data(BPoirier)
```

Format

A list with 8 components:
`$rect` is a vector of coordinates $(xmin, ymin, xmax, ymax)$ of the origin and the opposite corner of a 110 by 90 m rectangular plot.
`$tri1` is a list of vertex coordinates (ax, ay, bx, by, cx, cy) of contiguous triangles covering the denser part of the plot.
`$tri2` is a list of vertex coordinates (ax, ay, bx, by, cx, cy) of contiguous triangles covering the sparser part of the plot.
`$poly1` is a list of vertex coordinates (x, y) of the polygon enclosing `BPoirier$tri1`.
`$poly2` is a list of two polygons vertex coordinates (x, y) enclosing `BPoirier$tri2`.
`$trees` is a list of tree coordinates (x, y).
`$species` is a factor with 3 levels $("beech", "oak", "hornbeam")$ corresponding to species names of the trees.
`$dbh` is a vector of tree size (diameter at breast height in cm).

Source

Parde, J. 1981. De 1882 a 1976/80 : les places d'experience de sylviculture du hetre en foret domaniale de Haye. *Revue Forestiere Francaise*, 33: 41-64.

References

Goreaud, F. 2000. *Apports de l'analyse de la structure spatiale en foret temperee a l'etude et la modelisation des peuplements complexes*. These de doctorat, ENGREF, Nancy, France.

Pelissier, R. & Goreaud, F. 2001. A practical approach to the study of spatial structure in simple cases of heterogeneous vegetation. *Journal of Vegetation Science*, 12: 99-108.

Examples

```
data(BPoirier)
BP.spp <- spp(BPoirier$trees, mark=BPoirier$species, window=BPoirier$rect)
plot(BP.spp)
```

`Couepia` *Spatial pattern of Couepia caryophylloides in Paracou, a canopy tree species of French Guiana.*

Description

Spatial pattern of 34 mature individuals and 173 young individuals of the tree species *Couepia caryophylloides* (Chrysobalanaceae) in a 25-ha forest plot in Paracou, French Guiana.

Usage

```
data(Couepia)
```

Format

A list with 4 components:
`$rect` is a vector of coordinates $(xmin, ymin, xmax, ymax)$ of the origin and the opposite corner of a 500 by 500 m rectangular plot.
`$tri` is a list of vertex coordinates (ax, ay, bx, by, cx, cy) of contiguous triangles covering swampy parts of the plot.
`$trees` is a list of tree coordinates (x, y).
`$stage` is a factor with 2 levels $("mature", "young")$.

Source

Collinet, F. 1997. *Essai de regroupement des principales especes structurantes d'une foret dense humide d'apres l'analyse de leur repartition spatiale (foret de Paracou - Guyane).* These de doctorat, Universite Claude Bernard, Lyon, France.

References

Goreaud, F. & P?Pelissier, R. 2003. Avoiding misinterpretation of biotic interactions with the intertype *K12*-function: population independence vs. random labelling hypotheses. *Journal of Vegetation Science*, 14: 681-692.

Examples

```
data(Couepia)
coca.spp <- spp(Couepia$trees, mark=Couepia$stage, window=Couepia$rect, triangles=Couepia$tri)
plot(coca.spp)
```

---

demopat *Artificial Data Point Pattern from* spatstat.data *package.*

Description

This is an artificial dataset, for use in testing and demonstrating compatibility between spatstat and ads objects. It is a multitype point pattern in an irregular polygonal window. There are two types of points. The window contains a polygonal hole.

Usage

```
data(demopat)
```

Format

An object of class "ppp" representing a spatstat.core point pattern.

Source

data(demopat) in spatstat.data

Examples

```
data(demopat)
demo.spp<-ppp2spp(demopat)
plot(demo.spp)
```

dval *Multiscale local density of a spatial point pattern*

Description

Computes local density estimates of a spatial point pattern, i.e. the number of points per unit area, within sample circles of regularly increasing radii r, centred at the nodes of a grid covering a simple (rectangular or circular) or complex sampling window (see Details).

Usage

```
dval(p, upto, by, nx, ny)
```

Arguments

p	a "spp" object defining a spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nx, ny	number of sample circles regularly spaced out in x and y directions.

Details

The local density is estimated for a regular sequence of sample circles radii given by seq(by,upto,by) (see seq). The sample circles are centred at the nodes of a regular grid with size nx by ny. Ripley's edge effect correction is applied when the sample circles overlap boundary of the sampling window (see Ripley (1977) or Goreaud & P?Pelissier (1999) for an extension to circular and complex sampling windows). Due to edge effect correction, upto, the maximum radius of the sample circles, is half the longer side for a rectangle sampling window (i.e. $0.5 * max((xmax - xmin), (ymax - ymin))$) and the radius $r0$ for a circular sampling window (see swin).

Value

A list of class c("vads","dval") with essentially the following components:

r	a vector of regularly spaced out distances (seq(by,upto,by)).
xy	a data frame of $(nx * ny)$ observations giving (x, y) coordinates of the centres of the sample circles (the grid nodes).
cval	a matrix of size $(nx * ny, length(r))$ giving the estimated number of points of the pattern per sample circle with radius r.
dval	a matrix of size $(nx * ny, length(r))$ giving the estimated number of points of the pattern per unit area per sample circle with radius r.

Warning

In its current version, function dval ignores the marks of multivariate and marked point patterns (they are all considered to be univariate patterns).

Note

There are printing, summary and plotting methods for "vads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud, F. and P?Pelissier, R. 1999. On explicit formula of edge effect correction for Ripley's *K*-function. *Journal of Vegetation Science*, 10:433-438.

P?Pelissier, R. and Goreaud, F. 2001. A practical approach to the study of spatial structure in simple cases of heterogeneous vegetation. *Journal of Vegetation Science*, 12:99-108.

Ripley, B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-212.

See Also

plot.vads, spp.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
swr <- spp(BP$trees, win=BP$rect)
dswr <- dval(swr,25,1,11,9)
summary(dswr)
plot(dswr)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45))
dswc <- dval(swc,25,1,9,9)
summary(dswc)
plot(dswc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri1)
dswrt <- dval(swrt,25,1,11,9)
summary(dswrt)
plot(dswrt)
```

`inside.swin` *Test whether points are inside a sampling window*

Description

Function `inside.swin` tests whether points lie inside or outside a given sampling window.

Usage

```
inside.swin(x, y, w, bdry=TRUE)
```

Arguments

`x`	a vector of `x` coordinates of points.
`y`	a vector of `y` coordinates of points.
`w`	an object of class `"swin"` (see `swin`) defining the sampling window.
`bdry`	by default `bdry = TRUE`. If `FALSE`, points located on the boundary of the sampling window are considered to be outside.

Value

A logical vector whose `i`th entry is `TRUE` if the corresponding point $(x[i], y[i])$ is inside w, `FALSE` otherwise.

Note

For `"complex"` sampling windows, points inside the triangles to remove or on their boundary, are considered outside.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

`swin`.

Examples

```
data(BPoirier)
BP <- BPoirier
wr <- swin(BP$rect)
sum(inside.swin(BP$trees$x, BP$trees$y, wr))

wc <- swin(c(55,45,45))
sum(inside.swin(BP$trees$x, BP$trees$y, wc))

wrt <- swin(BP$rect, triangles=BP$tri1)
sum(inside.swin(BP$trees$x, BP$trees$y,wrt))
```

k12fun *Multiscale second-order neighbourhood analysis of a bivariate spatial point pattern*

Description

Computes estimates of the intertype *K12*-function and associated neighbourhood functions from a bivariate spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypotheses of population independence or random labelling (see Details).

Usage

```
k12fun(p, upto, by, nsim=0, H0=c("pitor","pimim","rl"), prec=0.01, nsimax=3000, conv=50,
 rep=10, alpha=0.01,  marks)
```

Arguments

p	a "spp" object defining a multivariate spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nsim	number of Monte Carlo simulations to estimate local confidence limits of the selected null hypothesis (see Details). By default nsim=0, so that no confidence limits are computed.
H0	one of c("pitor","pimim","rl") to select either the null hypothesis of population independence using toroidal shift (H0="pitor") or mimetic point process (H0="pimim"), or of random labelling (H0="rl") (see Details). By default, the null hypothesis is population independence using toroidal shift.
prec	if nsim>0 and H0="pitor" or H0="pimim", precision of the random vector or point coordinates generated during simulations. By default prec=0.01.
nsimax	if nsim>0 and H0="pimim", maximum number of simulations allowed (see mimetic. By default nsimax=3000.
conv	if nsim>0 and H0="pimim", convergence criterion (see mimetic. By default conv=50.
rep	if nsim>0 and H0="pimim", controls for convergence failure of the mimetic point process (see details). By default rep=10 so that the function aborts after 10 consecutive failures in mimetic point process convergence.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.
marks	by default c(1,2), otherwise a vector of two numbers or character strings identifying the types (the p$marks levels) of points of type 1 and 2, respectively.

Details

Function k12fun computes the intertype $K12(r)$ function of second-order neighbourhood analysis and the associated functions $g12(r)$, $n12(r)$ and $L12(r)$.

For a homogeneous isotropic bivariate point process of intensities $\lambda 1$ and $\lambda 2$, the second-order property could be characterized by a function $K12(r)$ (Lotwick & Silverman 1982), so that the expected number of neighbours of type 2 within a distance r of an arbitrary point of type 1 is: $N12(r) = \lambda 2 * K12(r)$.

$K12(r)$ is an intensity standardization of $N12(r)$: $K12(r) = N12(r)/\lambda 2$.

$n12(r)$ is an area standardization of of $N12(r)$: $n12(r) = N12(r)/(\pi * r^2)$, where $\pi * r^2$ is the area of the disc of radius r.

$L12(r)$ is a linearized version of $K12(r)$, which has an expectation of 0 under population independence: $L12(r) = \sqrt{(K12(r)/\pi)} - r$. $L12(r)$ becomes positive when the two population show attraction and negative when they show repulsion. Under the null hypothesis of random labelling, the expectation of $L12(r)$ is $L(r)$. It becomes greater than $L(r)$ when the types tend to be positively correlated and lower when they tend to be negatively correlated.

$g12(r)$ is the derivative of $K12(r)$ or bivariate pair density function, so that the expected number of points of type 2 at a distance r of an arbitrary point of type 1 (i.e. within an annuli between two successive circles with radii r and $r - by$) is: $O12(r) = \lambda 2 * g12(r)$ (Wiegand & Moloney 2004).

The program introduces an edge effect correction term according to the method proposed by Ripley (1977) and extended to circular and complex sampling windows by Goreaud & Pelissier (1999).

Theoretical values under the null hypothesis of either population independence or random labelling as well as local Monte Carlo confidence limits and p-values of departure from the null hypothesis (Besag & Diggle 1977) are estimated at each distance r.

The population independence hypothesis assumes that the location of points of a given population is independent from the location of points of the other. It is therefore tested conditionally to the intrinsic spatial pattern of each population. Two different procedures are available: `H0="pitor"` just shifts the pattern of type 1 points around a torus following Lotwick & Silverman (1982); `H0="pimim"` uses a mimetic point process (Goreaud et al. 2004) to mimic the pattern of type 1 points (see mimetic.
The random labelling hypothesis `"rl"` assumes that the probability to bear a given mark is the same for all points of the pattern and doesn't depends on neighbours. It is therefore tested conditionally to the whole spatial pattern, by randomizing the marks over the points' locations kept unchanged (see Goreaud & Pelissier 2003 for further details).

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced out distances (seq(by,upto,by)).
g12	a data frame containing values of the bivariate pair density function $g12(r)$.
n12	a data frame containing values of the bivariate local neighbour density function $n12(r)$.
k12	a data frame containing values of the intertype function $K12(r)$.
l12	a data frame containing values of the modified intertype function $L12(r)$.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected under the selected null hypothesis.
sup	(optional) if nsim>0 a vector of the upper local confidence limits of the selected null hypothesis at a significant level α.
inf	(optional) if nsim>0 a vector of the lower local confidence limits of the selected null hypothesis at a significant level α.
pval	(optional) if nsim>0 a vector of local p-values of departure from the selected null hypothesis.

Note

There are printing and plotting methods for "fads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Besag J.E. & Diggle P.J. 1977. Simple Monte Carlo tests spatial patterns. *Applied Statistics*, 26:327-333.

Goreaud F. & Pelissier R. 1999. On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, 10:433-438.

Goreaud, F. & Pelissier, R. 2003. Avoiding misinterpretation of biotic interactions with the intertype *K12*-function: population independence vs. random labelling hypotheses. *Journal of Vegetation Science*, 14: 681-692.

Lotwick, H.W. & Silverman, B.W. 1982. Methods for analysing spatial processes of several types of points. *Journal of the Royal Statistical Society B*, 44:403-413.

Ripley B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-192.

Wiegand, T. & Moloney, K.A. 2004. Rings, circles, and null-models for point pattern analysis in ecology. *Oikos*, 104:209-229. Goreaud F., Loussier, B., Ngo Bieng, M.-A. & Allain R. 2004. Simulating realistic spatial structure for forest stands: a mimetic point process. In *Proceedings of Interdisciplinary Spatial Statistics Workshop*, 2-3 December, 2004. Paris, France.

See Also

plot.fads, spp, k12val, kfun, kijfun, ki.fun, mimetic, kmfun.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
#testing population independence hypothesis
k12swrm.pi <- k12fun(swrm, 25, 1, 500, marks=c("beech","oak"))
plot(k12swrm.pi)
#testing random labelling hypothesis
k12swrm.rl <- k12fun(swrm, 25, 1, 500, H0="rl", marks=c("beech","oak"))
plot(k12swrm.rl)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45), marks=BP$species)
k12swc.pi <- k12fun(swc, 25, 1, 500, marks=c("beech","oak"))
plot(k12swc.pi)

## Not run: spatial point pattern in a complex sampling window
swrt.rl <- spp(BP$trees, win=BP$rect, tri=BP$tri2, marks=BP$species)
k12swrt.rl <- k12fun(swrt.rl, 25, 1, 500, H0="rl",marks=c("beech","oak"))
plot(k12swrt.rl)
## Not run: testing population independence hypothesis requires minimizing the outer polygon
xr<-range(BP$tri3$ax,BP$tri3$bx,BP$tri3$cx)
yr<-range(BP$tri3$ay,BP$tri3$by,BP$tri3$cy)
rect.min<-swin(c(xr[1], yr[1], xr[2], yr[2]))
swrt.pi <- spp(BP$trees, window = rect.min, triangles = BP$tri3, marks=BP$species)
k12swrt.pi <- k12fun(swrt.pi, 25, 1, nsim = 500, marks = c("beech", "oak"))
plot(k12swrt.pi)
```

---

k12val — *Multiscale local second-order neighbour density of a bivariate spatial point pattern*

Description

Computes local second-order neighbour density estimates for a bivariate spatial point pattern, i.e. the number of neighbours of type 2 per unit area within sample circles of regularly increasing radii r, centred at each type 1 point of the pattern (see Details).

Usage

```
k12val(p, upto, by, marks)
```

Arguments

p	a "spp" object defining a multivariate spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
marks	by default c(1,2), otherwise a vector of two numbers or character strings identifying the types (the p$marks levels) of points of type 1 and 2, respectively.

Details

Function K12val returns individual values of *K12(r)* and associated functions (see k12fun) estimated at each type 1 point of the pattern. For a given distance *r*, these values can be mapped within the sampling window, as in Getis & Franklin 1987 or P?Pelissier & Goreaud 2001.

Value

A list of class c("vads","k12val") with essentially the following components:

r	a vector of regularly spaced distances (seq(by,upto,by)).
xy	a data frame with 2 components giving (x, y) coordinates of type 1 points of the pattern.
g12val	a matrix of size $(length(xy), length(r))$ giving individual values of the bivariate pair density function $g12(r)$.
n12val	a matrix of size $(length(xy), length(r))$ giving individual values of the bivariate neighbour density function $n12(r)$.
k12val	a matrix of size $(length(xy), length(r))$ giving individual values of the intertype function $K12(r)$.
l12val	a matrix of size $(length(xy), length(r))$ giving individual values the modified intertype function $L12(r)$.

Note

There are printing, summary and plotting methods for "vads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Getis, A. and Franklin, J. 1987. Second-order neighborhood analysis of mapped point patterns. *Ecology*, 68:473-477.

P?Pelissier, R. and Goreaud, F. 2001. A practical approach to the study of spatial structure in simple cases of heterogeneous vegetation. *Journal of Vegetation Science*, 12:99-108.

See Also

plot.vads, k12fun, dval, kval.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
k12vswrm <- k12val(swrm, 25, 1, marks=c("beech","oak"))
summary(k12vswrm)
plot(k12vswrm)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45), marks=BP$species)
k12vswc <- k12val(swc, 25, 1, marks=c("beech","oak"))
summary(k12vswc)
plot(k12vswc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri2, marks=BP$species)
k12vswrt <- k12val(swrt, 25, 1, marks=c("beech","oak"))
summary(k12vswrt)
plot(k12vswrt)
```

---

kdfun — *Multiscale second-order neighbourhood analysis of a spatial phylogenetic or functional community pattern from fully mapped data*

Description

Computes distance-dependent estimates of Shen et al. (2014) phylogenetic or functional mark correlation functions from a multivariate spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypothesis of species equivalence (see Details).

Usage

```
kdfun(p, upto, by, dis, nsim=0, alpha = 0.01)
```

Arguments

p	a "spp" object defining a spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
dis	a "dist" object defining Euclidean distances between species.

nsim	number of Monte Carlo simulations to estimate local confidence limits of the null hypothesis of a random allocation of species distances (species equivalence; see Details). By default nsim = 0, so that no confidence limits are computed.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.

Details

Function kdfun computes Shen et al. (2014) Kd and gd-functions. For a multivariate point pattern consisting of S species with intensity λp, such functions can be estimated from the bivariate Kpq-functions between each pair of different species p and q. Function kdfun is thus a simple wrapper of k12fun (P?Pelissier & Goreaud 2014):

$Kd(r) = D{*}Kr(r)/HD{*}Ks(r) = D{*}sum(\lambda p{*}\lambda q{*}Kpq(r){*}dpq)/HD{*}sum(\lambda p{*}\lambda q{*}Kpq(r))$.
$gd(r) = D * g(r)/HD * gs(r) = D * sum(\lambda p * \lambda q * gpq(r) * dpq)/HD * sum(\lambda p * \lambda q * gpq(r))$.

where $Ks(r)$ and $gs(r)$ are distance-dependent versions of Simpson's diversity index, D (see ksfun), $Kr(r)$ and $gr(r)$ are distance-dependent versions of Rao's diversity coefficient (see krfun); dpq is the distance between species p and q defined by matrix dis, typically a taxonomic, phylogenetic or functional distance. The advantage here is that as the edge effects vanish between $Kr(r)$ and $Ks(r)$, implementation is fast for a sampling window of any shape. $Kd(r)$ provides the expected phylogenetic or functional distance of two heterospecific individuals a distance less than *r* apart (Shen et al. 2014), while $gd(r)$ provides the same within an annuli between two consecutive distances of *r* and *r-by*.

Theoretical values under the null hypothesis of species equivalence as well as local Monte Carlo confidence limits and p-values of departure from the null hypothesis (Besag & Diggle 1977) are estimated at each distance r, by randomizing the between-species distances, keeping the point locations and distribution of species labels unchanged. The theoretical expectations of $gd(r)$ and $Kd(r)$ are thus 1.

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced out distances (seq(by,upto,by)).
gd	a data frame containing values of the function $gd(r)$.
kd	a data frame containing values of the function $Kd(r)$.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected under the null hypothesis of species equivalence.
sup	(optional) if nsim>0 a vector of the upper local confidence limits of a random distribution of the null hypothesis at a significant level α.
inf	(optional) if nsim>0 a vector of the lower local confidence limits of a random distribution of the null hypothesis at a significant level α.

pval (optional) if nsim>0 a vector of local p-values of departure from the null hypothesis.

Note

There are printing and plotting methods for "fads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Shen, G., Wiegand, T., Mi, X. & He, F. (2014). Quantifying spatial phylogenetic structures of fully stem-mapped plant communities. *Methods in Ecology and Evolution*, 4, 1132-1141.

P?Pelissier, R. & Goreaud, F. ads package for R: A fast unbiased implementation of the K-function family for studying spatial point patterns in irregular-shaped sampling windows. *Journal of Statistical Software*, in press.

See Also

plot.fads, spp, ksfun, krfun, divc.

Examples

```
data(Paracou15)
P15<-Paracou15
## Not run: spatial point pattern in a rectangle sampling window of size 125 x 125
swmr <- spp(P15$trees, win = c(175, 175, 250, 250), marks = P15$species)
## Not run: testing the species equivalence hypothesis
kdswmr <- kdfun(swmr, dis = P15$spdist, 50, 2, 100)
## Not run: running more simulations is slow
kdswmr <- kdfun(swmr, dis = P15$spdist, 50, 2, 500)
plot(kdswmr)

## Not run: spatial point pattern in a circle with radius 50 centred on (125,125)
swmc <- spp(P15$trees, win = c(125,125,50), marks = P15$species)
kdswmc <- kdfun(swmc, dis = P15$spdist, 50, 2, 100)
## Not run: running more simulations is slow
kdswmc <- kdfun(swmc, dis = P15$spdist, 50, 2, 500)
plot(kdswmc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(P15$trees, win = c(125,125,250,250), tri = P15$tri, marks = P15$species)
kdswrt <- kdfun(swrt, dis = P15$spdist, 50, 2, 100)
## Not run: running simulations is slow
kdswrt <- kdfun(swrt, dis = P15$spdist, 50, 2, 500)
plot(kdswrt)
```

kfun *Multiscale second-order neighbourhood analysis of an univariate spatial point pattern*

Description

Computes estimates of Ripley's *K*-function and associated neighbourhood functions from an univariate spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypothesis of Complete Spatial Randomness (see Details).

Usage

```
kfun(p, upto, by, nsim=0, prec=0.01, alpha=0.01)
```

Arguments

p	a "spp" object defining a spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nsim	number of Monte Carlo simulations to estimate local confidence limits of the null hypothesis of complete spatial randomness (CSR) (see Details). By default nsim=0, so that no confidence limits are computed.
prec	if nsim>0, precision of points' coordinates generated during simulations. By default prec=0.01.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.

Details

Function kfun computes Ripley's $K(r)$ function of second-order neighbourhood analysis and the associated functions $g(r)$, $n(r)$ and $L(r)$.

For a homogeneous isotropic point process of intensity λ, Ripley (1977) showed that the second-order property could be characterized by a function $K(r)$, so that the expected number of neighbours within a distance r of an arbitrary point of the pattern is: $N(r) = \lambda * K(r)$.

$K(r)$ is a intensity standardization of $N(r)$, which has an expectation of $\pi * r^2$ under the null hypothesis of CSR: $K(r) = N(r)/\lambda$.

$n(r)$ is an area standardization of $N(r)$, which has an expectation of λ under the null hypothesis of CSR: $n(r) = N(r)/(\pi * r^2)$, where $\pi * r^2$ is the area of the disc of radius r.

$L(r)$ is a linearized version of $K(r)$ (Besag 1977), which has an expectation of 0 under the null hypothesis of CSR: $L(r) = \sqrt{(K(r)/\pi)} - r$. *L(r)* becomes positive when the pattern tends to clustering and negative when it tends to regularity.

$g(r)$ is the derivative of $K(r)$ or pair density function (Stoyan et al. 1987), so that the expected number of neighbours at a distance r of an arbitrary point of the pattern (i.e. within an annuli between two successive circles with radii r and $r - by$) is: $O(r) = \lambda * g(r)$.

The program introduces an edge effect correction term according to the method proposed by Ripley (1977) and extended to circular and complex sampling windows by Goreaud & P?Pelissier (1999).

Theoretical values under the null hypothesis of CSR as well as local Monte Carlo confidence limits and p-values of departure from CSR (Besag & Diggle 1977) are estimated at each distance r.

Value

A list of class `"fads"` with essentially the following components:

`r`	a vector of regularly spaced out distances (`seq(by,upto,by)`).
`g`	a data frame containing values of the pair density function $g(r)$.
`n`	a data frame containing values of the local neighbour density function $n(r)$.
`k`	a data frame containing values of Ripley's function $K(r)$.
`l`	a data frame containing values of the modified Ripley's function $L(r)$.

Each component except `r` is a data frame with the following variables:

`obs`	a vector of estimated values for the observed point pattern.
`theo`	a vector of theoretical values expected for a Poisson pattern.
`sup`	(optional) if `nsim>0` a vector of the upper local confidence limits of a Poisson pattern at a significant level α.
`inf`	(optional) if `nsim>0` a vector of the lower local confidence limits of a Poisson pattern at a significant level α.
`pval`	(optional) if `nsim>0` a vector of local p-values of departure from a Poisson pattern.

Warning

Function `kfun` ignores the marks of multivariate and marked point patterns, which are analysed as univariate patterns.

Note

There are printing and plotting methods for `"fads"` objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Besag J.E. 1977. Discussion on Dr Ripley's paper. *Journal of the Royal Statistical Society B*, 39:193-195.

Besag J.E. & Diggle P.J. 1977. Simple Monte Carlo tests spatial patterns. *Applied Statistics*, 26:327-333.

Goreaud F. & P?Pelissier R. 1999. On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, 10:433-438.

Ripley B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-192.

Stoyan D., Kendall W.S. & Mecke J. 1987. *Stochastic geometry and its applications*. Wiley, New-York.

See Also

plot.fads, spp, kval, k12fun, kijfun, ki.fun, kmfun.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
swr <- spp(BP$trees, win=BP$rect)
kswr <- kfun(swr,25,1,500)
plot(kswr)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45))
kswc <- kfun(swc, 25, 1, 500)
plot(kswc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri1)
kswrt <- kfun(swrt, 25, 1, 500)
plot(kswrt)
```

---

kmfun *Multiscale second-order neighbourhood analysis of a marked spatial point pattern*

Description

Computes estimates of the mark correlation *Km*-function and associated neighbourhood functions from a marked spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypothesis of no correlation between marks (see Details).

Usage

```
kmfun(p, upto, by, nsim=0, alpha=0.01)
```

Arguments

p	a "spp" object defining a marked spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nsim	number of Monte Carlo simulations to estimate local confidence limits of the null hypothesis of no correlation between marks (see Details). By default nsim=0, so that no confidence limits are computed.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.

Details

Function kmfun computes the mark correlation function $Km(r)$ and the associated function $gm(r)$.

It is defined from a general definition of spatial autocorrelation (Goreaud 2000) as:

$$Km(r) = (COV(Xi, Xj)|d(i,j) < r)/VAR(X)$$

where X is a quantitative random variable attached to each point of the pattern.

Km(r) has a very similar interpretation than more classical correlation functions, such as Moran's *I*: it takes values between -1 and 1, with an expectation of 0 under the null hypothesis of no spatial correlation between the values of *X*, becomes positive when values of X at distance *r* are positively correlated and negative when values of X at distance *r* are negatively correlated.

$gm(r)$ is the derivative of $Km(r)$ or pair mark correlation function, which gives the correlation of marks within an annuli between two successive circles with radii r and $r - by$).

The program introduces an edge effect correction term according to the method proposed by Ripley (1977) and extended to circular and complex sampling windows by Goreaud & P?Pelissier (1999).

Local Monte Carlo confidence limits and p-values of departure from the null hypothesis of no correlation are estimated at each distance r, after reallocating at random the values of *X* over all points of the pattern, the location of trees being kept unchanged.

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced out distances (seq(by,upto,by)).
gm	a data frame containing values of the pair mark correlation function $gm(r)$.
km	a data frame containing values of the mark correlation function $Km(r)$.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected for the null hypothesis of no correlation between marks.
sup	(optional) if nsim>0 a vector of the upper local confidence limits of the null hypothesis at a significant level α.
inf	(optional) if nsim>0 a vector of the lower local confidence limits of the null hypothesis at a significant level α.
pval	(optional) if nsim>0 a vector of local p-values of departure from the null hypothesis.

Note

Applications of this function can be found in Oddou-Muratorio *et al.* (2004) and Madelaine *et al.* (submitted).

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud, F. 2000. *Apports de l'analyse de la structure spatiale en foret tempere a l'etude et la modelisation des peuplements complexes*. These de doctorat, ENGREF, Nancy, France.

Goreaud F. & P?Pelissier R. 1999. On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, 10:433-438.

Madelaine, C., Pelissier, R., Vincent, G., Molino, J.-F., Sabatier, D., Prevost, M.-F. & de Namur, C. 2007. Mortality and recruitment in a lowland tropical rainforest of French Guiana: effects of soil type and species guild. *Journal of Tropical Ecology*, 23:277-287.

Oddou-Muratorio, S., Demesure-Musch, B., Pelissier, R. & Gouyon, P.-H. 2004. Impacts of gene flow and logging history on the local genetic structure of a scattered tree species, Sorbus torminalis L. *Molecular Ecology*, 13:3689-3702.

Ripley B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-192.

See Also

plot.fads, spp, kfun, k12fun, kijfun, ki.fun.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
```

```
swrm <- spp(BP$trees, win=BP$rect, marks=BP$dbh)
kmswrm <- kmfun(swrm, 25, 2, 500)
plot(kmswrm)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45), marks=BP$dbh)
kmswc <- kmfun(swc, 25, 2, 500)
plot(kmswc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri2, marks=BP$dbh)
kmswrt <- kmfun(swrt, 25, 2, 500)
plot(kmswrt)
```

---

kp.fun *Multiscale second-order neighbourhood analysis of a multivariate spatial point pattern*

Description

(Formerly ki.fun) Computes a set of *K12*-functions between all possible marks p and the other marks in a multivariate spatial point pattern defined in a simple (rectangular or circular) or complex sampling window (see Details).

Usage

```
kp.fun(p, upto, by)
```

Arguments

p	a "spp" object defining a multivariate spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).

Details

Function kp.fun is simply a wrapper to k12fun, which computes *K12(r)* between each mark p of the pattern and all other marks grouped together (the j points).

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced distances (seq(by,upto,by)).
labp	a vector containing the levels i of p$marks.
gp.	a data frame containing values of the pair density function $g12(r)$.

np.	a data frame containing values of the local neighbour density function $n12(r)$.
kp.	a data frame containing values of the $K12(r)$ function.
lp.	a data frame containing values of the modified $L12(r)$ function.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected under the null hypothesis of population independence (see k12fun).

Note

There are printing and plotting methods for "fads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

plot.fads, spp, kfun, k12fun, kpqfun.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: multivariate spatial point pattern in a rectangle sampling window
swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
kp.swrm <- kp.fun(swrm, 25, 1)
plot(kp.swrm)

## Not run: multivariate spatial point pattern in a circle with radius 50 centred on (55,45)
swcm <- spp(BP$trees, win=c(55,45,45), marks=BP$species)
kp.swcm <- kp.fun(swcm, 25, 1)
plot(kp.swcm)

## Not run: multivariate spatial point pattern in a complex sampling window
swrtm <- spp(BP$trees, win=BP$rect, tri=BP$tri2, marks=BP$species)
kp.swrtm <- kp.fun(swrtm, 25, 1)
plot(kp.swrtm)
```

kpqfun — *Multiscale second-order neighbourhood analysis of a multivariate spatial point pattern*

Description

(Formerly kijfun) Computes a set of *K*- and *K12*-functions for all possible pairs of marks (p, q) in a multivariate spatial point pattern defined in a simple (rectangular or circular) or complex sampling window (see Details).

Usage

```
kpqfun(p, upto, by)
```

Arguments

p	a "spp" object defining a multivariate spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).

Details

Function kpqfun is simply a wrapper to kfun and k12fun, which computes either *K(r)* for points of mark p when $p = q$ or *K12(r)* between the marks p and q otherwise.

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced distances (seq(by,upto,by)).
labpq	a vector containing the (p, q) paired levels of p$marks.
gpq	a data frame containing values of the pair density functions $g(r)$ and $g12(r)$.
npq	a data frame containing values of the local neighbour density functions $n(r)$ and $n12(r)$.
kpq	a data frame containing values of the $K(r)$ and $K12(r)$ functions.
lpq	a data frame containing values of the modified $L(r)$ and $L12(r)$ functions.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected under the null hypotheses of spatial randomness (see kfun) and population independence (see k12fun).

Note

There are printing and plotting methods for "fads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

plot.fads, spp, kfun, k12fun, kp.fun.

Examples

```
 data(BPoirier)
 BP <- BPoirier
## Not run: multivariate spatial point pattern in a rectangle sampling window
 swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
 kpqswrm <- kpqfun(swrm, 25, 1)
 plot(kpqswrm)

## Not run: multivariate spatial point pattern in a circle with radius 50 centred on (55,45)
 swcm <- spp(BP$trees, win=c(55,45,45), marks=BP$species)
 kpqswcm <- kpqfun(swcm, 25, 1)
 plot(kpqswcm)

 ## Not run: multivariate spatial point pattern in a complex sampling window
 swrtm <- spp(BP$trees, win=BP$rect, tri=BP$tri2, marks=BP$species)
 kpqswrtm <- kpqfun(swrtm, 25, 1)
 plot(kpqswrtm)
```

---

krfun — *Multiscale second-order neighbourhood analysis of a multivariate spatial point pattern using Rao quadratic entropy*

Description

Computes distance-dependent estimates of Rao's quadratic entropy from a multivariate spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypothesis of either a random labelling or a species equivalence (see Details).

Usage

```
krfun(p, upto, by, nsim=0, dis = NULL, H0 = c("rl", "se"), alpha = 0.01)
```

Arguments

p	a "spp" object defining a spatial point pattern in a given sampling window (see spp).
upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nsim	number of Monte Carlo simulations to estimate local confidence limits of the null hypothesis of a random allocation of species labels (see Details). By default nsim = 0, so that no confidence limits are computed.
dis	(optional) a "dist" object defining Euclidean distances between species. By default $dis = NULL$ so that species are considered equidistant.
H0	one of c("rl","se") to select either the null hypothesis of random labelling (H0 = "rl") or species equivalence (H0 = "se") (see Details). By default, the null hypothesis is random labelling.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.

Details

Function krfun computes distance-dependent functions of Rao (1982) quadratic entropy (see divc in package ade4).

For a multivariate point pattern consisting of S species with intensity λp, such functions can be estimated from the bivariate Kpq-functions between each pair of different species p and q. Function krfun is thus a simple wrapper function of k12fun and kfun, standardized by Rao diversity coefficient (Pelissier & Goreaud 2014):

$Kr(r) = sum(\lambda p * \lambda q * Kpq(r) * dpq)/(\lambda * \lambda * K(r) * HD)$.
$gr(r) = sum(\lambda p * \lambda q * gpq(r) * dpq)/(\lambda * \lambda * g(r) * HD)$.

where dpq is the distance between species p and q defined by matrix dis, typically a taxonomic, phylogenetic or functional distance, and $HD = sum(Np * Nq * dpq/(N(N-1)))$ is the unbiased version of Rao diversity coefficient (see Shimatani 2001). When dis = NULL, species are considered each other equidistant and krfun returns the same results than ksfun.

The program introduces an edge effect correction term according to the method proposed by Ripley (1977) and extended to circular and complex sampling windows by Goreaud & Pelissier (1999).

Theoretical values under the null hypothesis of either random labelling or species equivalence as well as local Monte Carlo confidence limits and p-values of departure from the null hypothesis (Besag & Diggle 1977) are estimated at each distance r.

The random labelling hypothesis (H0 = "rl") is tested by reallocating species labels at random among all points of the pattern, keeping the point locations unchanged, so that expectations of $gr(r)$ and $Kr(r)$ are 1 for all r. The species equivalence hypothesis (H0 = "se") is tested by randomizing the between-species distances, keeping the point locations and distribution of species labels unchanged. The theoretical expectations of $gr(r)$ and $Kr(r)$ are thus $gs(r)$ and $Ks(r)$, respectively (see ksfun).

Value

A list of class `"fads"` with essentially the following components:

`r`	a vector of regularly spaced out distances (`seq(by,upto,by)`).
`gr`	a data frame containing values of the function $gr(r)$.
`kr`	a data frame containing values of the function $Kr(r)$.

Each component except `r` is a data frame with the following variables:

`obs`	a vector of estimated values for the observed point pattern.
`theo`	a vector of theoretical values expected under the selected null hypothesis.
`sup`	(optional) if `nsim>0` a vector of the upper local confidence limits of a random distribution of the selected null hypothesis at a significant level α.
`inf`	(optional) if `nsim>0` a vector of the lower local confidence limits of a random distribution of the selected null hypothesis at a significant level α.
`pval`	(optional) if `nsim>0` a vector of local p-values of departure from the selected null hypothesis.

Note

There are printing and plotting methods for `"fads"` objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Rao, C.R. 1982. Diversity and dissimilarity coefficient: a unified approach. *Theoretical Population Biology*, 21:24-43.

Shimatani, K. 2001. On the measurement of species diversity incorporating species differences. *Oikos*, 93, 135-147.

Goreaud F. & Pelissier R. 1999. On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, 10:433-438.

Ripley B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-192.

Pelissier, R. & Goreaud, F. 2014. ads package for R: A fast unbiased implementation of the k-function family for studying spatial point patterns in irregular-shaped sampling windows. *Journal of Statistical Software*, in press.

See Also

`plot.fads`, `spp`, `ksfun`, `kdfun`, `divc`.

Examples

```
data(Paracou15)
P15<-Paracou15
## Not run: spatial point pattern in a rectangle sampling window of size 125 x 125
swmr <- spp(P15$trees, win = c(175, 175, 250, 250), marks = P15$species)
## Not run: testing the random labeling hypothesis
krwmr.rl <- krfun(swmr, dis = P15$spdist, H0 = "rl", 25, 2, 50)
## Not run: running more simulations is slow
krwmr.rl <- krfun(swmr, dis = P15$spdist, H0 = "rl", 25, 2, 500)
plot(krwmr.rl)
## Not run: testing the species equivalence hypothesis
krwmr.se <- krfun(swmr, dis = P15$spdist, H0 = "se", 25, 2, 50)
## Not run: running more simulations is slow
krwmr.se <- krfun(swmr, dis = P15$spdist, H0 = "se", 25, 2, 500)
plot(krwmr.se)

## Not run: spatial point pattern in a circle with radius 50 centred on (125,125)
swmc <- spp(P15$trees, win = c(125,125,50), marks = P15$species)
krwmc <- krfun(swmc, dis = P15$spdist, H0 = "rl", 25, 2, 100)
## Not run: running more simulations is slow
krwmc <- krfun(swmc, dis = P15$spdist, H0 = "rl", 25, 2, 500)
plot(krwmc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(P15$trees, win = c(125,125,250,250), tri = P15$tri, marks = P15$species)
krwrt <- krfun(swrt, dis = P15$spdist, H0 = "rl", 25, 2)
## Not run: running simulations is slow
krwrt <- krfun(swrt, dis = P15$spdist, H0 = "rl", 25, 2, 500)
plot(krwrt)
```

`ksfun` *Multiscale second-order neighbourhood analysis of a multivariate spatial point pattern using Simpson diversity*

Description

Computes estimates of Shimatani *alpha* and *beta* functions of Simpson diversity from a multivariate spatial point pattern in a simple (rectangular or circular) or complex sampling window. Computes optionally local confidence limits of the functions under the null hypothesis of a random allocation of species labels (see Details).

Usage

```
ksfun(p, upto, by, nsim=0, alpha=0.01)
```

Arguments

p	a "spp" object defining a spatial point pattern in a given sampling window (see spp).

upto	maximum radius of the sample circles (see Details).
by	interval length between successive sample circles radii (see Details).
nsim	number of Monte Carlo simulations to estimate local confidence limits of the null hypothesis of a random allocation of species labels (see Details). By default nsim=0, so that no confidence limits are computed.
alpha	if nsim>0, significant level of the confidence limits. By default $\alpha = 0.01$.

Details

Function ksfun computes Shimatani $\alpha(r)$ and $\beta(r)$ functions of Simpson diversity, called here $Ks(r)$ and $gs(r)$, respectively.

For a multivariate point pattern consisting of S species with intensity λp, Shimatani (2001) showed that a distance-dependent measure of Simpson (1949) diversity can be estimated from Ripley (1977) K-function computed for each species separately and for all the points grouped together (see also Eckel et al. 2008). Function ksfun is thus a simple wrapper function of kfun, standardized by Simpson diversity coefficient:

$Ks(r) = 1 - sum(\lambda p * \lambda p * Kp(r))/(\lambda * \lambda * K(r) * D)$ which is a standardized estimator of $\alpha(r)$ in Shimatani (2001).

$gs(r) = 1 - sum(\lambda p * \lambda p * gp(r))/(\lambda * \lambda * g(r) * D)$ corresponding to a standardized version of $\beta(r)$ in Shimatani (2001).

$Kp(r)$ and $K(r)$ (resp. $gp(r)$ and $g(r)$) are univariate K-functions computed for species p and for all species together; $D = 1 - sum(Np * (Np-1)/(N * (N-1)))$ is the unbiased version of Simpson diversity, with Np the number of individuals of species p in the sample and $N = sum(Np)$.

The program introduces an edge effect correction term according to the method proposed by Ripley (1977) and extended to circular and complex sampling windows by Goreaud & P?Pelissier (1999).

The theoretical values of $gr(r)$ and $Kr(r)$ under the null hypothesis of random labelling is 1 for all r. Local Monte Carlo confidence limits and p-values of departure from this hypothesis are estimated at each distance r by reallocating at random the species labels among points of the pattern, keeping the point locations unchanged.

Value

A list of class "fads" with essentially the following components:

r	a vector of regularly spaced out distances (seq(by,upto,by)).
gs	a data frame containing values of the function $gs(r)$.
ks	a data frame containing values of the function $Ks(r)$.

Each component except r is a data frame with the following variables:

obs	a vector of estimated values for the observed point pattern.
theo	a vector of theoretical values expected under the null hypothesis of random labelling, i.e. 1 for all r.
sup	(optional) if nsim>0 a vector of the upper local confidence limits of a random distribution of species labels at a significant level α.
inf	(optional) if nsim>0 a vector of the lower local confidence limits of a random distribution of species labels at a significant level α.
pval	(optional) if nsim>0 a vector of local p-values of departure from a random distribution of species labels.

Note

There are printing and plotting methods for "fads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Shimatani K. 2001. Multivariate point processes and spatial variation in species diversity. *Forest Ecology and Management*, 142:215-229.

Eckel, S., Fleisher, F., Grabarnik, P. and Schmidt V. 2008. An investigation of the spatial correlations for relative purchasing power in Baden-Wurttemberg. *AstA - Advances in Statistical Analysis*, 92:135-152.

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

Goreaud F. & P?Pelissier R. 1999. On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, 10:433-438.

Ripley B.D. 1977. Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39:172-192.

See Also

plot.fads, spp, kfun, kpqfun, kp.fun, krfun.

Examples

```
data(Paracou15)
P15<-Paracou15
## Not run: spatial point pattern in a rectangle sampling window of size 125 x 125
swmr <- spp(P15$trees, win = c(125, 125, 250, 250), marks = P15$species)
kswmr <- ksfun(swmr, 50, 5, 500)
plot(kswmr)

## Not run: spatial point pattern in a circle with radius 50 centred on (125,125)
swmc <- spp(P15$trees, win = c(125, 125, 50), marks = P15$species)
kswmc <- ksfun(swmc, 50, 5, 500)
plot(kswmc)
```

```
## Not run: spatial point pattern in a complex sampling window
swrt <- spp(P15$trees, win = c(125, 125, 250, 250), tri=P15$tri, marks=P15$species)
kswrt <- ksfun(swrt, 50, 5, 500)
plot(kswrt)
```

`kval` *Multiscale local second-order neighbour density of a spatial point pattern*

Description

Computes local second-order neighbour density estimates for an univariate spatial point pattern, i.e. the number of neighbours per unit area within sample circles of regularly increasing radii r, centred at each point of the pattern (see Details).

Usage

```
kval(p, upto, by)
```

Arguments

`p`	a `"spp"` object defining a spatial point pattern in a given sampling window (see `spp`).
`upto`	maximum radius of the sample circles (see Details).
`by`	interval length between successive sample circles radii (see Details).

Details

Function `kval` returns individual values of *K(r)* and associated functions (see `kfun`) estimated for each point of the pattern. For a given distance *r*, these values can be mapped within the sampling window (Getis & Franklin 1987, P?Pelissier & Goreaud 2001).

Value

A list of class `c("vads","kval")` with essentially the following components:

`r`	a vector of regularly spaced out distances (`seq(by,upto,by)`).
`xy`	a data frame with 2 components giving (x, y) coordinates of points of the pattern.
`gval`	a matrix of size $(length(xy), length(r))$ giving individual values of the pair density function $g(r)$.
`nval`	a matrix of size $(length(xy), length(r))$ giving individual values of the neighbour density function $n(r)$.
`kval`	a matrix of size $(length(xy), length(r))$ giving individual values of Ripley's function $K(r)$.
`lval`	a matrix of size $(length(xy), length(r))$ giving individual values the modified Ripley's function $L(r)$.

Warning

Function kval ignores the marks of multivariate and marked point patterns (they are all considered to be univariate patterns).

Note

There are printing, summary and plotting methods for "vads" objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Getis, A. and Franklin, J. 1987. Second-order neighborhood analysis of mapped point patterns. *Ecology*, 68:473-477.

P?Pelissier, R. and Goreaud, F. 2001. A practical approach to the study of spatial structure in simple cases of heterogeneous vegetation. *Journal of Vegetation Science*, 12:99-108.

See Also

plot.vads, kfun, dval, k12val.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: spatial point pattern in a rectangle sampling window of size [0,110] x [0,90]
swr <- spp(BP$trees, win=BP$rect)
kvswr <- kval(swr, 25, 1)
summary(kvswr)
plot(kvswr)

## Not run: spatial point pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,45))
kvswc <- kval(swc, 25, 1)
summary(kvswc)
plot(kvswc)

## Not run: spatial point pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri1)
kvswrt <- kval(swrt, 25, 1)
summary(kvswrt)
plot(kvswrt)
```

mimetic *Univariate point pattern simulation by mimetic point process*

Description

Simulates replicates of an observed univariate point pattern by stochastic optimization of its L-function properties.

Usage

```
mimetic(x,upto=NULL,by=NULL,prec=NULL,nsimax=3000,conv=50)
```

Arguments

x	either a ("fads", "kfun") object or a "spp" object of type "univariate" defining a spatial point pattern in a given sampling window (see kfun or spp).
upto	(optional) maximum radius of the sample circles when x is a "spp" object.
by	(optional) interval length between successive sample circles radii when x is a "spp" object.
prec	precision of point coordinates generated during simulations when x is a "spp" object. By default prec=0.01 or the value used in function kfun when x is a ("fads", "kfun") object.
nsimax	maximum number of simulations allowed. By default the process stops after nsimax=3000 if convergence is not reached.
conv	maximum number of simulations without optimization gain (convergence criterion).

Details

Function mimetic uses a stepwise depletion-replacement algorithm to generate a point pattern whose L-function is optimized with regards to an observed one, following the mimetic point process principle (Goreaud et al. 2004). Four points are randomly deleted at each step of the process and replaced by new points that minimize the following cost function:$||Lobs(r) - Lsim(r)||$)^2. The simulation stops as soon as the cost function doesn't decrease after conv simulations or after a maximum of nsimax simulations. The process apply to rectangular, circular or complex sampling windows (see spp). There exist a plot method that displays diagnostic plots, i.e. the observed and simulated L-function, the simulated point pattern and the successive values of the cost function.

Value

A list of class "mimetic" with essentially the following components:

call	the function call.
fads	an object of class ("fads", "mimetic") with 2 components:

`..r`	a vector of regularly spaced out distances corresponding to seq(by,upto,by).
`..l`	a dataframe with 2 components:
`....obs`	a vector of values of the L-function estimated for the initial observed pattern
`....sim`	a vector of values of the L-function estimated for the simulated pattern
`spp`	a object of class `"spp"` corresponding to the simulated point pattern (see spp).
`theo`	a vector of theoretical values, i.e. Simpson D for all the points.
`cost`	a vector of the successive values of the cost function.

Note

There are printing and plotting methods for `"mimetic"` objects.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud F., Loussier, B., Ngo Bieng, M.-A. & Allain R. 2004. Simulating realistic spatial structure for forest stands: a mimetic point process. In Proceedings of Interdisciplinary Spatial Statistics Workshop, 2-3 December, 2004. Paris, France.

See Also

spp, kfun,

Examples

```
data(BPoirier)
BP<-BPoirier
## Not run: performing point pattern analysis in a rectangle sampling window
swr <- spp(BP$trees, win=BP$rect)
plot(swr)

## Not run: performing the mimetic point process from "spp" object
mimswr <- mimetic(swr, 20, 2)
plot(mimswr)

## Not run: performing the mimetic point process from "fads" object
mimkswr <- mimetic(kfun(swr, 20, 2))
plot(mimkswr)
```

Paracou15 *Tree spatial pattern in control plot 15, Paracou experimental station, French Guiana*

Description

Spatial pattern of 4128 trees of 332 different species in a 250 m X 250 m control plot in Paracou experimental station, French Guiana.

Usage

```
data(Paracou15)
```

Format

A list with 5 components:
`$rect` is a vector of coordinates $(xmin, ymin, xmax, ymax)$ of the origin and the opposite corner of a 250 by 250 m rectangular plot.
`$trees` is a list of tree coordinates (x, y).
`$species` is a factor with 332 levels corresponding to species names of the trees.
`$spdist` is an object of class `"dist"` giving between-species distances based on functional traits (see Paine et al. 2011).

Source

Gourlet-Fleury, S., Ferry, B., Molino, J.-F., Petronelli, P. & Schmitt, L. 2004. *Experimental plots: key features.* Pp. 3-60 In Gourlet-Fleury, S., Guehl, J.-M. & Laroussinie, O. (Eds.), Ecology and Management of a Neotropical rainforest - Lessons drawn from Paracou, a long-term experimental research site in French Guiana. Elsevier SAS, France.

References

Paine, C. E. T., Baraloto, C., Chave, J. & Herault, B. 2011. Functional traits of individual trees reveal ecological constraints on community assembly in tropical rain forests. *Oikos*, 120: 720-727.

Examples

```
data(Paracou15)
P15.spp <- spp(Paracou15$trees, mark = Paracou15$species, window = Paracou15$rect)
plot(P15.spp, chars = rep("o", 332), cols = rainbow(332), legend = FALSE, maxsize = 0.5)
```

plot.fads *Plot second-order neighbourhood functions*

Description

Plot second-order neighbourhood function estimates returned by functions kfun, k12fun, kmfun, kijfun or ki.fun.

Usage

```
## S3 method for class 'fads'
plot(x, opt, cols, lty, main, sub, legend, csize, ...)
```

Arguments

x	an object of class "fads" (see Details).
opt	one of c("all","L","K","n","g") to display either all or one of the functions in a single window. By default opt = "all" for fads objects of subclass "kfun", "k12fun", or "kmfun"; by default opt = "L" for fads objects of subclass "kij", or "ki.".
cols	(optional) colours used for plotting functions.
lty	(optional) line types used for plotting functions.
main	by default, the value of argument x, otherwise a text to be displayed as a title of the plot. main=NULL displays no title.
sub	by default, the name of the function displayed, otherwise a text to be displayed as function subtitle. sub=NULL displays no subtitle.
legend	If legend = TRUE (the default) a legend for the plotting functions is displayed.
csize	scaling factor for font size so that actual font size is par("cex")*csize. By default csize = 1.
...	extra arguments that will be passed to the plotting functions plot.swin, plot.default, symbols and/or points.

Details

Function plot.fads displays second-order neighbourhood function estimates as a function of inter-point distance, with expected values as well as confidence interval limits when computed. Argument x can be any fads object returned by functions kfun, k12fun, kmfun, kijfun or ki.fun.

Value

none.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

kfun, k12fun, kmfun, kijfun, ki.fun.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: Ripley's function
swr <- spp(BP$trees, win=BP$rect)
k.swr <- kfun(swr, 25, 1, 500)
plot(k.swr)

## Not run: Intertype function
swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
k12.swrm <- k12fun(swrm, 25, 1, 500, marks=c("beech","oak"))
plot(k12.swrm, opt="L", cols=1)

## Not run: Mark correlation function
swrm <- spp(BP$trees, win=BP$rect, marks=BP$dbh)
km.swrm <- kmfun(swrm, 25, 1, 500)
plot(km.swrm, main="Example 1", sub=NULL, legend=FALSE)
```

---

plot.spp *Plot a Spatial Point Pattern object*

Description

Plot a Spatial Point Pattern object returned by function spp.

Usage

```
## S3 method for class 'spp'
plot(x, main, out=FALSE, use.marks=TRUE, cols, chars, cols.out, chars.out,
maxsize, scale=TRUE, add=FALSE, legend=TRUE, csize=1, ...)
```

Arguments

x	an object of class "spp" (see spp).
main	by default, the value of argument x, otherwise a text to be displayed as a title of the plot.main=NULL displays no title.
out	by default out = FALSE. If TRUE points of the pattern located outside the sampling window are plotted.
use.marks	by default use.marks = TRUE. If FALSE different symbols are not used for each mark of multivariate or marked point patterns, so that they are plotted as univariate (see spp).

cols	(optional) the colour(s) used to plot points located inside the sampling window (see Details).
chars	(optional) plotting character(s) used to plot points located inside the sampling window (see Details).
cols.out	(optional) if out = TRUE, the colour(s) used to plot points located outside the sampling window (see Details).
chars.out	(optional) if out = TRUE, plotting character(s) used to plot points located outside the sampling window (see Details).
maxsize	(optional) maximum size of plotting symbols. By default maxsize is automatically adjusted to plot size.
csize	scaling factor for font size so that actual font size is par("cex")*csize. By default csize = 1.
scale	If scale = TRUE (the default) graduations giving plot size are displayed.
legend	If legend = TRUE (the default) a legend for plot symbols is displayed (multivariate and marked types only).
add	by default add = FALSE. If TRUE a new window is not created and just the points are plotted over the existing plot.
...	extra arguments that will be passed to the plotting functions plot.default, points and/or symbols.

Details

The sampling window x$window is plotted first, through a call to function plot.swin. Then the points themselves are plotted, in a fashion that depends on the type of spatial point pattern (see spp).

- **univariate pattern:** if x$type = c("univariate"), i.e. the point pattern does not have marks, or if use.marks = FALSE, then the locations of all points is plotted using a single plot character.
- **multivariate pattern:** if x$type = c("multivariate"), i.e. the marks are levels of a factor, then each level is represented by a different plot character.
- **marked pattern:** if x$type = c("marked"), i.e. the marks are real numbers, then points are represented by circles (argument chars = "circles", the default) or squares (argument chars = "squares") proportional to their marks' value (positive values are filled, while negative values are unfilled).

Arguments cols and cols.out (if out = TRUE) determine the colour(s) used to display the points located inside and outside the sampling window, respectively. Colours may be specified as codes or colour names (see par("col")). For univariate and marked point patterns, cols and cols.out are single character strings, while for multivariate point patterns they are character vectors of same length as levels(x$marks) and levels(x$marksout), respectively.

Arguments chars and chars.out (if out = TRUE) determine the symbol(s) used to display the points located inside and outside the sampling window, respectively. Symbols may be specified as codes or character strings (see par("pch")). For univariate point patterns, chars and chars.out are single character strings, while for multivariate point patterns they are character vectors of same length as levels(x$marks) and levels(x$marksout), respectively. For marked point patterns, chars and chars.out can only take the value "circles" or "squares".

Value

none.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

spp, swin, plot.swin.

Examples

```
 data(BPoirier)
 BP<-BPoirier

 ## Not run: a univariate point pattern in a rectangle sampling window
 plot(spp(BP$trees, win=BP$rect))

 ## Not run: a univariate point pattern in a circular sampling window
plot(spp(BP$trees, win=c(55,45,45)), out=TRUE, scale=TRUE)

## Not run: a univariate point pattern in a complex sampling window
## Not run: (points outside the sampling window displayed in red colour)
plot(spp(BP$trees, win=BP$rect, tri=BP$tri1), out=TRUE)

## Not run: a multivariate point pattern in a rectangle sampling window
plot(spp(BP$trees, win=BP$rect, marks=BP$species))

## Not run: a multivariate point pattern in a circular sampling window
## Not run: (points inside/outside the sampling window displayed in blue colour/as red crosses)
plot(spp(BP$trees, win=c(55,45,45), marks=BP$species), out=TRUE, cols=c("blue","blue","blue"),
chars.out=c("+","+","+"), cols.out=c("red","red","red"))

 ## Not run: a marked point pattern in a rectangle sampling window with circles in green colour
 plot(spp(BP$trees, win=BP$rect, marks=BP$dbh), cols="green")

 ## Not run: a marked point pattern in a circular sampling window
 ## Not run: (squares in red colour inside and circles in blue colour outside)
 plot(spp(BP$trees, win=c(55,45,45), marks=BP$dbh), out=TRUE, chars="squares",
 cols="red", cols.out="blue")
```

---

plot.vads *Plot local density values*

Description

Plot local density estimates returned by functions dval, kval or k12val.

Usage

```
## S3 method for class 'vads'
plot(x, main, opt, select, chars, cols, maxsize, char0, col0, legend, csize, ...)
```

Arguments

x	an object of class 'vads' (see Details).
main	by default, the value of argument x, otherwise a text to be displayed as a title of the plot. main=NULL displays no title.
opt	(optional) a character string to change the type of values to be plotted (see Details).
select	(optional) a vector of selected distances in x$r. By default, a multiple window displays all distances.
chars	one of c("circles","squares") plotting symbols with areas proportional to local density values. By default, circles are plotted.
cols	(optional) the colour used for the plotting symbols. Black colour is the default.
maxsize	(optional) maximum size of the circles/squares plotted. By default, maxsize is automatically adjusted to plot size.
char0	(optional) the plotting symbol used to represent null values. By default, null values are not plotted.
col0	(optional) the colour used for the null values plotting symbol. By default, the same as argument cols.
legend	If legend = TRUE (the default) a legend for the plotting values is displayed.
csize	scaling factor for font size so that actual font size is par("cex")*csize. By default csize = 1.
...	extra arguments that will be passed to the plotting functions plot.swin, plot.default, symbols and/or points.

Details

Function plot.vads displays a map of first-order local density or second-order local neighbour density values as symbols with areas proportional to the values estimated at the plotted points. Positive values are represented by coloured symbols, while negative values are represented by open symbols. The plotted function values depend upon the type of 'vads' object:

- if class(x)=c("vads","dval"), the plotted values are first-order local densities and argument opt="dval" by default, but is potentially one of c("dval","cval") returned by dval.
- if class(x)=c("vads","kval") or class(x)=c("vads","k12val"), the plotted values are univariate or bivariate second-order local neighbour densities. Argument opt="lval" by default, but is potentially one of c("lval","kval","nval","gval") returned by kval and k12val.

Value

none.

Author(s)

<Raphael.Pelissier@ird.fr>

See Also

dval, kval, k12val.

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: local density in a rectangle sampling window
dswr <- dval(spp(BP$trees, win=BP$rect), 25, 1, 11, 9)
plot(dswr)
## Not run: display only distance r from 5 to 10 with null symbols as red crosses
plot(dswr, select=c(5:10), char0=3, col0="red")

## Not run: local L(r) values in a circular sampling window
lvswc <- kval(spp(BP$trees, win=c(55,45,45)), 25, 0.5)
plot(lvswc)
## Not run: display square symbols in blue for selected values of r and remove title
plot(lvswc, chars="squares", cols="blue", select=c(5,7.5,10,12.5,15), main=NULL)

## Not run: local K12(r) values (1="beech", 2="oak") in a complex sampling window
k12swrt <- k12val(spp(BP$trees, win=BP$rect, tri=BP$tri1, marks=BP$species), 25, 1)
plot(k12swrt, opt="kval")
```

---

spp *Creating a spatial point pattern*

Description

Function spp creates an object of class "spp", which represents a spatial point pattern observed in a finite sampling window (or study region). The ads library supports univariate, multivariate and marked point patterns observed in simple (rectangular or circular) or complex sampling windows.

Usage

```
spp(x, y=NULL, window, triangles, marks, int2fac=TRUE)
ppp2spp(p)
```

Arguments

x, y	if y=NULL, x is a list of two vectors of point coordinates, else both x and y are atomic vectors of point coordinates.
window	a "swin" object or a vector defining the limits of a simple sampling window: c(xmin,ymin,xmax,ymax) for a rectangle ; c(x0,y0,r0) for a circle.

`triangles`	(optional) a list of triangles removed from a simple initial window to define a complex sampling window (see `swin`).
`marks`	(optional) a vector of mark values, which may be factor levels or numerical values (see Details).
`int2fac`	if TRUE, integer marks are automatically coerced into factor levels.
`p`	a `"ppp"` object from package `spatstat.geom`.

Details

A spatial point pattern is assumed to have been observed within a specific sampling window (a finite study region) defined by the `window` argument. If `window` is a simple `"swin"` object, it may be coerced into a complex type by adding a `triangles` argument (see `swin`). A spatial point pattern may be of 3 different types.

- **univariate pattern:** by default when argument `marks` is not given.
- **multivariate pattern:** `marks` is a factor, which levels are interpreted as categorical marks (e.g. colours, species, etc.) attached to points of the pattern. Integer marks may be automatically coerced into factor levels when argument `int2fac = TRUE`.
- **marked pattern:** `marks` is a vector of real numbers attached to points of the pattern. Integer values may also be considered as numerical values if argument `int2fac = FALSE`.

Value

An object of class `"spp"` describing a spatial point pattern observed in a given sampling window.

`$type`	a character string indicating if the spatial point pattern is `"univariate"`, `"multivariate"` or `"marked"`.
`$window`	an `swin` object describing the sampling window (see `swin`).
`$n`	an integer value giving the number of points of the pattern located inside the sampling window (points on the boundary are considered to be inside).
`$x`	a vector of x coordinates of points located inside the sampling window.
`$y`	a vector of y coordinates of points located inside the sampling window.
`$nout`	(optional) an integer value giving the number of points of the pattern located outside the sampling window.
`$xout`	(optional) a vector of x coordinates of points located outside the sampling window.
`$yout`	(optional) a vector of y coordinates of points located outside the sampling window.
`$marks`	(optional) a vector of the marks attached to points located inside the sampling window.
`$marksout`	(optional) a vector of the marks attached to points located outside the sampling window.

Note

There are printing, summary and plotting methods for `"spp"` objects.
Function `ppp2spp` converts an `ppp.object` from package `spatstat.geom` into an `"spp"` object.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud, F. and P?Pelissier, R. 1999. On explicit formula of edge effect correction for Ripley's *K*-function. *Journal of Vegetation Science*, 10:433-438.

See Also

plot.spp, swin

Examples

```
data(BPoirier)
BP <- BPoirier
## Not run: univariate pattern in a rectangle of size [0,110] x [0,90]
swr <- spp(BP$trees, win=BP$rect)
## Not run: an alternative using atomic vectors of point coordinates
swr <- spp(BP$trees, win=BP$rect)
summary(swr)
plot(swr)

## Not run: univariate pattern in a circle with radius 50 centred on (55,45)
swc <- spp(BP$trees, win=c(55,45,50))
summary(swc)
plot(swc)
plot(swc, out=TRUE) # plot points outside the circle

## Not run: multivariate pattern in a rectangle of size [0,110] x [0,90]
swrm <- spp(BP$trees, win=BP$rect, marks=BP$species)
summary(swrm)
plot(swrm)
plot(swrm, chars=c("b","h","o")) # replace symbols by letters

## Not run: marked pattern in a rectangle of size [0,110] x [0,90]
swrn <- spp(BP$trees, win=BP$rect, marks=BP$dbh)
summary(swrn)
plot(swrn)

## Not run: multivariate pattern in a complex sampling window
swrt <- spp(BP$trees, win=BP$rect, tri=BP$tri1, marks=BP$species)
summary(swrt)
plot(swrt)
plot(swrt, out=TRUE) # plot points outside the sampling window


## Not run: converting a ppp object from spatstat.geom
data(demopat)
demo.spp<-ppp2spp(demopat)
plot(demo.spp)
```

swin *Creating a sampling window*

Description

Function swin creates an object of class "swin", which represents the sampling window (or study region) in which a spatial point pattern was observed. The ads library supports simple (rectangular or circular) and complex sampling windows.

Usage

```
swin(window, triangles)
owin2swin(w)
```

Arguments

window	a vector defining the limits of a simple sampling window: c(xmin,ymin,xmax,ymax) for a rectangle ; c(x0,y0,r0) for a circle.
triangles	(optional) a list of triangles removed from a simple initial window to define a complex sampling window (see Details).
‘	
w	a "owin" object from package spatstat.geom.

Details

A sampling window may be of simple or complex type. A simple sampling window may be a rectangle or a circle. A complex sampling window is defined by removing triangular surfaces from a simple (rectangular or circular) initial sampling window.

- **rectangular window:** window=c(ximn,ymin,xmax,ymax) a vector of length 4 giving the coordinates $(ximn, ymin)$ and $(xmax, ymax)$ of the origin and the opposite corner of a rectangle.
- **circular window:** window=c(x0,y0,r0) a vector of length 3 giving the coordinates $(x0, y0)$ of the centre and the radius $r0$ of a circle.
- **complex window:** triangles is a list of 6 variables giving the vertices coordinates (ax, ay, bx, by, cx, cy) of the triangles to remove from a simple (rectangular or circular) initial window. The triangles may be removed near the boundary of a rectangular window in order to design a polygonal sampling window, or within a rectangle or a circle, to delineating holes in the initial sampling window (see Examples). The triangles do not overlap each other, nor overlap boundary of the initial sampling window. Any polygon (possibly with holes) can be decomposed into contiguous triangles using triangulate.

Value

An object of class "swin" describing the sampling window. It may be of four different types with different arguments:

$type	a vector of two character strings defining the type of sampling window among c("simple","rectangle"), c("simple","circle"), c("complex","rectangle") or c("complex","circle").
$xmin, $ymin, $xmax, $ymax	(optional) coordinates of the origin and the opposite corner for a rectangular sampling window (see details).
$x0, $y0, $r0	(optional) coordinates of the centre and radius for a circular sampling window (see details).
$triangles	(optional) vertices coordinates of triangles for a complex sampling window (see details).

Note

There are printing, summary and plotting methods for "swin" objects.
Function owin2swin converts an owin.object from package spatstat.geom into an "swin" object.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud, F. and P?Pelissier, R. 1999. On explicit formula of edge effect correction for Ripley's *K*-function. *Journal of Vegetation Science*, 10:433-438.

See Also

area.swin, inside.swin, spp

Examples

```
## Not run: rectangle of size [0,110] x [0,90]
wr <- swin(c(0,0,110,90))
summary(wr)
plot(wr)

## Not run: circle with radius 50 centred on (55,45)
wc <- swin(c(55,45,50))
summary(wc)
plot(wc)

## Not run: polygon (diamond shape)
t1 <- c(0,0,55,0,0,45)
t2 <- c(55,0,110,0,110,45)
t3 <- c(0,45,0,90,55,90)
```

```
t4 <- c(55,90,110,90,110,45)
wp <- swin(wr, rbind(t1,t2,t3,t4))
summary(wp)
plot(wp)

## Not run: rectangle with a hole
h1 <- c(25,45,55,75,85,45)
h2 <- c(25,45,55,15,85,45)
wrh <- swin(wr, rbind(h1,h2))
summary(wrh)
plot(wrh)

## Not run: circle with a hole
wch <- swin(wc, rbind(h1,h2))
summary(wch)
plot(wch)

## Not run: converting an owin object from spatstat.geom
data(demopat)
demo.swin<-owin2swin(demopat$window)
plot(demo.swin)
```

`triangulate` *Triangulate polygon*

Description

Function `triangulate` decomposes a simple polygon (optionally having holes) into contiguous triangles.

Usage

```
triangulate(outer.poly, holes)
```

Arguments

`outer.poly`	a list with two component vectors `x` and `y` giving vertex coordinates of the polygon or a vector `(xmin,ymin,xmax,ymax)` giving coordinates $(ximn, ymin)$ and $(xmax, ymax)$ of the origin and the opposite corner of a rectangle sampling window (see `swin`).
`holes`	(optional) a list (or a list of list) with two component vectors `x` and `y` giving vertices coordinates of inner polygon(s) delineating hole(s) within the `outer.poly`.

Details

In argument `outer.poly`, the vertices must be listed following boundary of the polygon without any repetition (i.e. do not repeat the first vertex). Argument `holes` may be a list of vertices coordinates of a single hole (i.e. with x and y component vectors) or a list of list for multiple holes, where each `holes[[i]]` is a list with x and y component vectors. Holes' vertices must all be inside the

`outer.poly` boundary (vertices on the boundary are considered outside). Multiple holes do not overlap each others.

Value

A list of 6 variables, suitable for using in `swin` and `spp`, and giving the vertices coordinates (ax, ay, bx, by, cx, cy) of the triangles that pave the polygon. For a polygon with *t* holes totaling n vertices (outer contour + holes), the number of triangles produced is $(n-2)+2t$, with $n < 200$ in this version of the program.

Author(s)

<Raphael.Pelissier@ird.fr>

References

Goreaud, F. and P?Pelissier, R. 1999. On explicit formula of edge effect correction for Ripley's *K*-function. *Journal of Vegetation Science*, 10:433-438.

Narkhede, A. & Manocha, D. 1995. Fast polygon triangulation based on Seidel's algorithm. Pp 394-397 In A.W. Paeth (Ed.) *Graphics Gems V*. Academic Press. http://www.cs.unc.edu/~dm/CODE/GEM/chapter.html.

See Also

`spp`, `swin`

Examples

```
data(BPoirier)
BP <- BPoirier
plot(BP$poly1$x, BP$poly1$y)

## Not run: a single polygon triangulation
tri1 <- triangulate(BP$poly1)
plot(swin(BP$rect, tri1))

## Not run: a single polygon with a hole
tri2 <- triangulate(c(-10,-10,120,100), BP$poly1)
plot(swin(c(-10,-10,120,100), tri2))
```

Index