# Package ‘ecespa’

July 4, 2015

**Type** Package

**Title** Functions for Spatial Point Pattern Analysis

**Version** 1.1-8

**Date** 2015-06-23

**Author** Marcelino de la Cruz Rot, with contributions of Philip M. Dixon and Jose M. Blanco-Moreno

**Maintainer** Marcelino de la Cruz Rot <marcelino.delacruz@upm.es>

**Depends** splancs, spatstat (>= 1.4.0)

**Description** Some wrappers, functions and data sets for for spatial point pattern analysis (mainly based on spatstat), used in the book ``Introduccion al Analisis Espacial de Datos en Ecologia y Ciencias Ambientales: Metodos y Aplicaciones''.

**License** GPL (>= 2)

**LazyLoad** yes

**LazyData** yes

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2015-07-04 08:42:13

## R topics documented:

## `dixon2002` *Dixon (2002) Nearest-neighbor contingency table analysis*

### Description

`dixon2002` is a wrapper to the functions of Dixon (2002) to test spatial segregation for several species by analyzing the counts of the nearest neighbour contingency table for a marked point pattern.

### Usage

```
dixon2002(datos, nsim = 99)
```

### Arguments

| | |
|---|---|
| `datos` | `data.frame` with three columns: x-coordinate, y-coordinate and sp-name. See `swamp`. |
| `nsim` | number of simulations for the randomization approximation of the p-values. |

### Details

A measure of segregation describes the tendency of one species to be associated with itself or with other species. Dixon (2002) proposed a measure of the **segregation of species** *i* in a multiespecies spatial pattern as:

$$S[i] = log[(N[ii]/(N[i] - N[ii])]/[(N[i] - 1)/(N - N[i])]$$

where $N[i]$ is the number of individuals of species *i*, $N[ii]$ is the frequency of species *i* as neighbor of especies $i$ and $N$ is the total number of locations. Values of $S[i]$ larger than 0 indicate that species *i* is segregated; the larger the value of $S[i]$, the more extreme the segregation. Values of $S[i]$ less than 0 indicate that species *i* is is found as neighbor of itself less than expected under random labelling. Values of $S[i]$ close to 0 are consistent with random labelling of the neighbors of species *i*.

Dixon (2002) also proposed a **pairwise segregation index** for the off-diagonal elements of the contingency table:

$$S[ij] = log[(N[ij]/(N[i] - N[ij]))]/[(N[i])/(N - N[j]) - 1]$$

$S[ij]$ is larger than 0 when $N[ij]$, the frequency of neighbors of species *j* around points of species *i*, is larger than expected under random labelling and less than 0 when $N[ij]$ is smaller than expected under random labelling.

As a **species/neighbor-specific test**, Dixon(2002) proposed the statistic

$$Z[ij] = (N[ij] - EN[ij])/sqrt(VarN[ij])$$

where *j* may be the same as *i* and $EN[ij]$ is the expected count in the contingency table. It has an asymptotic normal distribution with mean 0 and variance 1; its asymptotic p-valure can be obtained from the numerical evaluation of the cumulative normal distribution or by simulation, i.e, by condicting a randomization test (appropriate when the sample size is small).

An **overall test of random labelling** (i.e. a test that all counts in the $k$ x $k$ nearest-neighbor contingency table are equal to their expected counts) is based on the quadratic form

$$C = (N - EN)'Sigma^{-}(N - EN)$$

where $N$ is the vector of all cell counts in the contingency table, $Sigma$ is the variance-covariance matrix of those counts and $Sigma^{-}$ is a generalized inverse of $Sigma$. Under the null hypothesis of random labelling of points, $C$ has a asymptotic Chi-square distribution with $k(k-1)$ degrees of freedom (if the sample sizes are small its distribution should be estimated using Monte-Carlo simulation). P-values are computed from the probability of observing equal or larger values of $C$. The overall statistic $C$ can be partitioned into $k$ **species-specific test** statistics $C[i]$. Each $C[i]$ test if the frequencies of the neighbors of species *i* are similar to the expected frequencies if the points were randomly labelled. Because the $C[i]$ are not independent Chi-square statistics, they do not sum to the overall $C$.

## Value

A list with the following components:

| | |
|---|---|
| `ON` | Observed nearest neighbor counts in table format. From row sp to column sp. |
| `EN` | Expected nearest neighbor counts in table format. |
| `Z` | Z-score for testing whether the observed count equals the expected count. |
| `S` | Segregation measure. |
| `pZas` | P-values based on the asymptotic normal distribution of the Z statistic. |
| `pZr` | If nsim !=0, p-values of the Z-score based on the randomization distribution. |
| `C` | Overall test of random labelling. |
| `Ci` | Species-specific test of random labelling. |
| `pCas` | P-value of the overall test from the asymptotic chi-square distribution with the appropriate degrees of freedom. |

| | |
|---|---|
| `pCias` | P-values of the species-specific tests from the asymptotic chi-square distribution with the appropriate degrees of freedom. |
| `pCr` | If nsim !=0, p-value of the overall test from the randomization distribution. |
| `pCir` | If nsim !=0, p-values of the species-specific tests from the randomization distribution. |
| `tablaZ` | table with ON, EN, Z, S, pZas and pZr in pretty format, as in the table II of Dixon (2002). |
| `tablaC` | table with C, Ci, pCas,pCias, pCr and pCir in pretty format, as in the table IV of Dixon (2002). |

## Warning

The $S[i]$ and $S[ij]$ statistics asume that the spatial nearest-neighbor process is stationary, at least to second order, i.e., have the same sign in every part of the entire plot. A biologically heterogeneous process will violate this asumption.

## Author(s)

Philip M. Dixon `http://www.public.iastate.edu/~pdixon/`.

Marcelino de la Cruz `<marcelino.delacruz@upm.es>` wrote the wrapper code for the `ecespa` version.

## References

Dixon, P.M. 2002. Nearest-neighbor contingency table analysis of spatial segregation for several species. *Ecoscience*, **9** (2): 142-151.

## See Also

`K012` for another segregation test, based in the differences of univariate and bivariate $K$-functions.

## Examples

```
## Not run:

data(swamp)

dixon2002(swamp,nsim=99)


## End(Not run)
```

## `ecespa` *Functions for spatial point pattern analysis in ecology*

### Description

Some wrappers, functions and data sets for spatial point pattern analysis, with an ecological bias.

### Details

| | |
|---|---|
| Package: | ecespa |
| Type: | Package |
| Version: | 1.1-3 |
| Date: | 2009-01-09 |
| License: | GPL (>=2) |

### Author(s)

Marcelino de la Cruz Rot, with contributions of Philip M. Dixon and Jose M. Blanco-Moreno and heavily borrowing Baddeley's & Turner's **spatstat** code.

Mantainer: Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>

### References

De la Cruz, M. 2008. Métodos para analizar datos puntuales. En: *Introducción al Análisis Espacial de Datos en Ecología y Ciencias Ambientales: Métodos y Aplicaciones* (eds. Maestre, F. T., Escudero, A. y Bonet, A.), pp 76-127. Asociación Española de Ecología Terrestre, Universidad Rey Juan Carlos y Caja de Ahorros del Mediterráneo, Madrid.

De la Cruz, M., Romao, R.L., Escudero, A. and Maestre, F.T. 2008. Where do seedlings go? A spatio-temporal analysis of early mortality in a semiarid specialist. *Ecography*,31 DOI: 10.1111/j.2008.0906-7590.05299 .

Diggle, P. J. 2003. *Statistical analysis of spatial point patterns*. Arnold, London.

Dixon, P.M. 2002. Nearest-neighbor contingency table analysis of spatial segregation for several species. *Ecoscience*, **9** (2): 142-151.

Olano, J.M., Laskurain, N.A., Escudero, A. and De la Cruz, M. 2009. Why and where adult trees die in a secondary temperate forest? The role of neighbourhood. *Annals of Forest Science* DOI: 10.1051/forest:2008074 .

Penttinen, A. 2006. Statistics for Marked Point Patterns. In *The Yearbook of the Finnish Statistical Society*, pp. 70-91.

Rey-Benayas, J.M., de la Montaña, E., Pérez-Camacho, L., de la Cruz, M., Moreno, D., Parejo, J.L. and Suárez-Seoane, S. 2008. Inter-annual dynamics and spatial congruence of a nocturnal bird assemblage inhabiting a Mediterranean agricultural mosaic. *Submitted.*

Syrjala, S. E. 1996. A statistical test for a difference between the spatial distributions of two populations. *Ecology* 77: 75-80.

## Examples

```
## Not run:

#############################################
### Transfom easily data from a data.frame into the ppp format
### of spatstat:

data(fig1)

plot(fig1) #typical xyplot

fig1.ppp <- haz.ppp (fig1)

fig1.ppp

plot(fig1.ppp) # point pattern plot of spatstat



#############################################
###  Summarize the joint pattern of points and marks at different scales
###  with the normalized mark-weighted K-function (Penttinen, 2006).
###  Compare this function in two consecutive cohorts of Helianthemum
###  squamatum seedlings:

 ## Figure 3.10 of De la Cruz (2008):

  data(seedlings1)

  data(seedlings2)

  s1km <- Kmm(seedlings1, r=1:100)

  s2km <- Kmm(seedlings2, r=1:100)

  plot(s1km, ylime=c(0.6,1.2), lwd=2, maine="", xlabe="r(cm)")

  plot(s2km,  lwd=2, lty=2, add=T )

  abline(h=1, lwd=2, lty=3)

  legend(x=60, y=1.2, legend=c("Hs_C1", "Hs_C2", "H0"),
 lty=c(1, 2, 3), lwd=c(3, 2, 2), bty="n")

## A pointwise test of normalized Kmm == 1 for seedlings1:

   s1km.test <- Kmm(seedlings1, r=1:100, nsim=99)

   plot(s1km.test,  xlabe="r(cm)")
```

```
#############################################
###  Explore the local relationships between marks and locations (e.g. size
###  of one cohort of H. squamatum seedlings). Map the marked point pattern
###  to a random field for visual inspection, with the normalized mark-sum
###  measure (Penttinen, 2006).

data(seedlings1)

 seed.m <- marksum(seedlings1, R=25)

 plot(seed.m, what="marksum", sigma = 5)  # raw mark-sum measure; sigma is bandwith for smoothing

 plot(seed.m, what="pointsum", sigma = 5) # point sum measure

 plot(seed.m,  what="normalized", dimyx=200, contour=TRUE, sigma = 5) # normalized  mark-sum measure

# the same with added grid

 plot(seed.m,  what="normalized", dimyx=200, contour=TRUE, sigma = 5, grid=TRUE)



#############################################
###  Test against the null model of "independent labelling",
###  i.e. test asociation/repulsion between  a "fixed" pattern (e.g. adult
###  H. squamatum plants) and a "variable" pattern (e.g. of surviving and
###  dead seedlings), with 2.5% and 97.5% envelopes of 999 random
###  labellings (De la Cruz & al. 2008).

data(Helianthemum)


cosa <- K012(Helianthemum, fijo="adultHS", i="deadpl", j="survpl",
             r=seq(0,200,le=201), nsim=999, nrank=25, correction="isotropic")

plot(cosa$k01, sqrt(./pi)-r~r,  col=c(3, 1, 3), lty=c(3, 1, 3), las=1,
         ylab=expression(L[12]), xlim=c(0, 200),
 main="adult HS vs. dead seedlings", legend=FALSE)

plot(cosa$k02, sqrt(./pi)-r~r, col=c(3, 1, 3), lty=c(3, 1, 3), las=1,
         ylab=expression(L[12]), xlim=c(0, 200),
   main="adult HS vs. surviving seedlings", legend=FALSE)



#############################################
###  Test differences of agregation and segregation between two patterns,
###  e.g. surviving and dying H. squamatum seedlings (De la Cruz & al. 2008).
```

```
data(Helianthemum)

cosa12 <- K1K2(Helianthemum, j="deadpl", i="survpl", r=seq(0,200,le=201),
 nsim=999, nrank=1, correction="isotropic")

plot(cosa12$k1k2, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
         main= "survival- death",ylab=expression(K[1]-K[2]), legend=FALSE)

plot(cosa12$k1k12, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
 main="segregation of surviving seedlings",ylab=expression(K[1]-K[12]), legend=FALSE)

plot(cosa12$k2k12, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
       main= "segregation of dying seedlings",ylab=expression(K[2]-K[12]), legend=FALSE)


#############################################
###  Test 'univariate' and 'bivariate' point patterns
###  against non-Poisson (in-)homogeneous models
###  (De la Cruz and Escudero, submited).

 data(urkiola)

   #####################
   ## univariate example

   # get univariate pp
   I.ppp <- split.ppp(urkiola)$birch

   # estimate inhomogeneous intensity function
   I.lam <- predict (ppm(I.ppp, ~polynom(x,y,2)), type="trend", ngrid=200)

   # Compute and plot envelopes to Kinhom, simulating from an Inhomogeneous
   #  Poisson Process:

   I2.env <- envelope( I.ppp,Kinhom, lambda=I.lam, correction="trans",
                             nsim=99, simulate=expression(rpoispp(I.lam)))
  plot(I2.env, sqrt(./pi)-r~r, xlab="r (metres)", ylab= "L (r)", col=c(1,3,2,2),legend=FALSE)

   # It seems that there is short scale clustering; let's fit an Inhomogeneous
   # Poisson Cluster Process:

   I.ki <- ipc.estK(mippp=I.ppp, lambda=I.lam, correction="trans")

   # Compute and plot envelopes to Kinhom, simulating from the fitted IPCP:

   Ipc.env <- Ki(I.ki, correction="trans", nsim=99, ngrid=200)

   plot (Ipc.env, xlab="r (metres)", ylab= "L (r)")

   #####################
   ## bivariate example: test independence between birch and quercus in Urkiola

   J.ppp <- split.ppp(urkiola)$oak
   # We want to simulate oak from a homogeneous Poisson model:
   J.ppm <- ppm(J.ppp, trend=~1, interaction=Poisson() )

   IJ.env <- Kci (mod1=I.ki, mod2=J.ppm, nsim=99)

   plot(IJ.env, type=12)

   plot(IJ.env, type=21)




#############################################
###  Simulate envelopes from the fitted values of a logistic model,
###  as in Olano et al. (2009)


   data(quercusvm)

   # read fitted values from logistic model:


   probquercus <-c(0.99955463, 0.96563477, 0.97577094, 0.97327199, 0.92437309,
   0.84023396, 0.94926682, 0.89687281, 0.99377915, 0.74157478, 0.95491518,
   0.72366493, 0.66771787, 0.77330148, 0.67569082, 0.9874892, 0.7918891,
   0.73246803, 0.81614635, 0.66446411, 0.80077908, 0.98290508, 0.54641754,
   0.53546689, 0.73273626, 0.7347013, 0.65559655, 0.89481468, 0.63946334,
   0.62101995, 0.78996371, 0.93179582, 0.80160346, 0.82204428, 0.90050059,
   0.83810669, 0.92153079, 0.47872421, 0.24697004, 0.50680935, 0.6297911,
   0.46374812, 0.65672284, 0.87951682, 0.35818237, 0.50932432, 0.92293014,
   0.48580241, 0.49692053, 0.52290553, 0.7317549, 0.32445982, 0.30300865,
   0.73599359, 0.6206056, 0.85777043, 0.65758613, 0.50100406, 0.31340849,
   0.22289286, 0.40002879, 0.29567678, 0.56917817, 0.56866864, 0.27718552,
   0.4910667, 0.47394411, 0.40543788, 0.29571349, 0.30436276, 0.47859015,
   0.31754526, 0.42131675, 0.37468782, 0.73271225, 0.26786274, 0.59506388,
   0.54801851, 0.38983575, 0.64896835, 0.37282031, 0.67624306, 0.29429766,
   0.29197755, 0.2247629, 0.40697843, 0.17022391, 0.26528042, 0.24373722,
   0.26936163, 0.13052254, 0.19958585, 0.18659692, 0.36686678, 0.47263005,
   0.39557661, 0.68048997, 0.74878567, 0.88352322, 0.93851375)



   ################################
   ## Envelopes for an homogeneous point pattern:

   cosap <- Kinhom.log(A=quercusvm, lifemark="0", nsim=99, prob=probquercus)

   plot(cosap)


   ################################
   ## Envelopes for an inhomogeneous point pattern:
```

```
   ## First, fit an inhomogeneous Poisson model to alive trees :

   quercusalive <- unmark(quercusvm[quercusvm$marks == 0])

    mod2 <- ppm(quercusalive, ~polynom(x,y,2))

    ## Now use mod2 to estimate lambda for K.inhom:

    cosapm <- Kinhom.log(A=quercusvm, lifemark="0", prob=probquercus,
                                 nsim=99,  mod=mod2)
    plot(cosapm)




#############################################
###  Test segregation based on the counts in the contingency table
###  of nearest neighbors in a multitype point pattern (Dixon, 2002)

data(swamp)

dixon2002(swamp,nsim=99)





#############################################
###  Fit the Poisson cluster point process to a point pattern with
###  the method of minimum contrast (Diggle 2003).

data(gypsophylous)

# Estimate K function ("Kobs").

gyps.env <- envelope(gypsophylous, Kest, correction="iso", nsim=99)

plot(gyps.env, sqrt(./pi)-r~r, ylab="L(r)", legend=FALSE)

# Fit Poisson Cluster Process. The limits of integration
# rmin and rmax are setup to 0 and 60, respectively.

cosa.pc <- pc.estK(Kobs = gyps.env$obs[gyps.env$r<=60],
           r = gyps.env$r[gyps.env$r<=60])

# Add fitted Kclust function to the plot.

lines(gyps.env$r,sqrt(Kclust(gyps.env$r, cosa.pc$sigma2,cosa.pc$rho)/pi)-gyps.env$r,
       lty=2, lwd=3, col="purple")

# A kind of pointwise test of the gypsophylous pattern been a realisation
# of the fitted model, simulating with sim.poissonc and using function J (Jest).
```

```
gyps.env.sim <- envelope(gypsophylous, Jest, nsim=99,
                    simulate=expression(sim.poissonc(gypsophylous,
    sigma=sqrt(cosa.pc$sigma2), rho=cosa.pc$rho)))

 plot(gyps.env.sim,  main="",legendpos="bottomleft")





#############################################
###  Compute Syrjala's test for the difference between the spatial
###  distributions of two populations, as in Rey-Benayas et al.
### (submited)



   data(syr1); data(syr2); data(syr3)

   plot(syrjala.test(syr1, syr2, nsim=999))

   plot(syrjala.test(syr1, syr3, nsim=999))





## End(Not run)
```

---

`figuras` *Artificial point data.*

---

## Description

The three different point patterns in the figure 3.1 of De la Cruz (2008)

## Usage

```
data(fig1)
data(fig2)
data(fig3)
```

## Format

A data frame with 87 observations on the following 2 variables.

x x coordinate

y y coordinate

## References

De la Cruz, M. 2008. Métodos para analizar datos puntuales. En: *Introducción al Análisis Espacial de Datos en Ecología y Ciencias Ambientales: Métodos y Aplicaciones* (eds. Maestre, F. T., Escudero, A. y Bonet, A.), pp 76-127. Asociación Española de Ecología Terrestre, Universidad Rey Juan Carlos y Caja de Ahorros del Mediterráneo, Madrid.

## Examples

```
## Not run:
data(fig1)

data(fig2)

data(fig3)

# transform to ppp format of spatstat with function haz.ppp:

fig1.ppp <- haz.ppp(fig1)

fig2.ppp <- haz.ppp(fig2)

fig3.ppp <- haz.ppp(fig3)

#Analyses as in Fig.3.2 of De la Cruz (2008). First, compute function K:

cosa1 <- Kest(fig1.ppp)

# Plot different estimators.
# Fig. 3.2a:

par("mar"=par("mar")+c(0,1,0,0))

plot(cosa1, col=c(1,0,0,1), lwd=c(2,2,2,2), lty=c(1,1,1,2),
 main="")

# Fig. 3.2b:

plot(cosa1, sqrt(./pi)-r~r, col=c(1,0,0,1), lwd=c(2,2,2,2),
         lty=c(1,1,1,2), main="", ylab="L(r)")

# Fig. 3.2c:

plot(cosa1, .-(pi*r^2)~r, col=c(1,0,0,1), lwd=c(2,2,2,2),
         lty=c(1,1,1,2), main="", ylab=expression(K(r)-pi*r^2))

# Fig. 3.2d:

plot(cosa1,(./(pi*r^2))-1~r, col=c(1,0,0,1), lwd=c(2,2,2,2),
 lty=c(1,1,1,2), main="",
         ylab=expression((K(r)/pi*r^2)-1))

## Analyses as in fig. 3.7 of De la Cruz (2008).
## First, compute function K and pointwise envelopes:

cosa1.env <- envelope(fig1.ppp, Kest)

cosa2.env <- envelope(fig2.ppp, Kest)

cosa3.env <- envelope(fig3.ppp, Kest)

## Plot function L with pointwise envelopes:

plot(cosa1.env,sqrt(./pi)-r~r, lwd=c(1,1,2,2),
 lty=c(1,1,3,3), col=c(1,1,1,1), xlab="r",
         ylab="L(r)", main="", ylim=c(-2,2))

## Add simultaneous envelopes of Ripley (+-1.68 *sqrt(A)/N):

abline(h=1.68*sqrt(fig1.ppp$w$area)/fig1.ppp$n,
         lty=2, lwd=2)

abline(h=-1.68*sqrt(fig1.ppp$w$area)/fig1.ppp$n,
         lty=2, lwd=2)

## Plot function L with pointwise envelopes:

plot(cosa2.env,sqrt(./pi)-r~r, lwd=c(1,1,2,2),
         lty=c(1,1,3,3), col=c(1,1,1,1), xlab="r",
         ylab="L(r)", main="")

## Add simultaneous envelopes of Ripley:

abline(h=1.68*sqrt(fig2.ppp$w$area)/fig2.ppp$n,
         lty=2, lwd=2)

abline(h=-1.68*sqrt(fig2.ppp$w$area)/fig2.ppp$n,
         lty=2, lwd=2)

## Plot function L with pointwise envelopes:

plot(cosa3.env,sqrt(./pi)-r~r, lwd=c(1,1,2,2),
         lty=c(1,1,3,3), col=c(1,1,1,1), xlab="r",
         ylab="L(r)", main="")

## Add simultaneous envelopes of Ripley:

abline(h=1.68*sqrt(fig3.ppp$w$area)/fig3.ppp$n,
         lty=2, lwd=2)

abline(h=-1.68*sqrt(fig3.ppp$w$area)/fig3.ppp$n,
         lty=2, lwd=2)

## End(Not run)
```

| | |
|---|---|
| getis | *Neighbourhood density function* |

## Description

Computes and plots the neighbourhood density function, a local version of the $K$-function defined by Getis and Franklin (1987).

## Usage

```
getis(mippp, nx = 30, ny = 30, R = 10)

## S3 method for class 'ecespa.getis'
plot(x, type="k", dimyx=NULL, xy=NULL, eps=NULL,  color=NULL,
         contour=TRUE, points=TRUE,...)
```

## Arguments

| | |
|---|---|
| `mippp` | A point pattern. An object with the ppp format of `spatstat`. |
| `nx` | Grid dimensions (for estimation) in the x-side. |
| `ny` | Grid dimensions (for estimation) in the y-side. |
| `R` | Radius. The distance argument *r* at which the function $K$ should be computed. |
| `x` | Result of applying `getis` to a point pattern. |
| `type` | Type of local statistics to be ploted. One of `k` (local-$K$), `l` (local-$L$), `n` (local-$n$) or `d` (deviations from CSR). |
| `color` | A list of colors such as that generated by `rainbow`, `heat.colors`, `topo.colors`, `terrain.colors` or similar functions. |
| `dimyx` | pixel array dimensions, will be passed to (i.e. see details in) `as.mask`. |
| `xy` | pixel coordinates, will be passed to (i.e. see details in) `as.mask`. |
| `eps` | width and height of pixels, will be passed to (i.e. see details in) `as.mask`. |
| `contour` | Logical; if TRUE, add a contour to current plot. |
| `points` | Logical; if TRUE, add the point pattern to current plot. |
| `...` | Additional graphical parameters passed to `link{plot}`. |

## Details

Getis and Franklin (1987) proposed the neigbourhood density function, a local version of Ripley's $L$- function. Given a spatial point pattern $X$, the neigbourhood density function associated with the $i$th point in $X$ is computed by

$$L[i](r) = sqrt((a/((n-1))*pi))*sum[j]e[i,j])$$

where the sum is over all points *j != i* that lie within a distance *r* of the *i*th point, *a* is the area of the observation window, *n* is the number of points in $X$, and *e[i,j]* is the isotropic edge correction term (as described in `Kest`). The value of *L[i](r)* can also be interpreted as one of the summands that contributes to the global estimate of the $L$-function.

The command `getis` actually computes the local $K$-function using `Kcross`. As the main objective of `getis` is to map the local density function, as sugested by Gestis and Franklin (1987: 476) a grid of points (whose density is controled by `nx` and `ny`), is used to accurately estimate the functions in empty or sparse areas. The S3 method `plot.ecespa.getis` plots the spatial distribution of the local $K$ or $L$ function or other related local statistics, such as $n[i](r)$, the number of neighbor points [$=lambda * K[i](r)$] or the deviations from the expected value of local $L$ under CSR [$= L[i](r) - r$]. It some of the arguments `dimyx`, `xy` or `eps` is provided it will use the function `interp.im` in `spatstat` package to interpolate the results;otherwise it will plot the estimated values at the origial grid points.

## Value

`getis` gives an object of class `ecespa.getis`, bassically a list with the following elements:

| | |
|---|---|
| `x` | x coordinates of pattern points (ahead) and grid points. |
| `y` | y coordinates of pattern points (ahead) and grid points. |
| `klocal` | Estimate of local $K[i](r)$ at the point pattern points. |
| `klocalgrid` | Estimate of local $K[i](r)$ at the grid points. |
| `R` | Distance $r$ at which the estimation is made. |
| `nx` | Density of the estimating grid in the x-side. |
| `ny` | Density of the estimating grid in the x-side. |
| `dataname` | Name of the ppp object analysed. |
| `ppp` | Original point pattern. |

`plot.ecespa.getis` plots an interpolated map of the selected local statistics

## Note

As `plot.ecespa.getis` interpolates over rectangular grid of points, it is not apropriate to map irregular windows. In those cases, `smooth.ppp` of `spatstat` can be used to interpolate the local statistics (see examples).

## Author(s)

Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>

## References

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

## See Also

`localK`, a different approach in **spatstat**.

## Examples

```
 ## Not run:
  ## Compare with fig. 5b of Getis & Franklin (1987: 476):

  data(ponderosa)

  ponderosa12 <- getis(ponderosa, nx = 30, ny = 30, R = 12)

  plot(ponderosa12, type = "l", dimyx=256)

  ## Plot the same, using smooth.ppp in spatstat

  ponderosa.12 <- setmarks(ponderosa, ponderosa12$klocal)

  Z <- Smooth(ponderosa.12, sigma=5, dimyx=256)

  plot(Z, col=topo.colors(128), main="smoothed neighbourhood density")

  contour(Z, add=TRUE)

  points(ponderosa, pch=16, cex=0.5)

  ## Example with irregular window:

  data(letterR)

  X <- rpoispp(50, win=letterR)

  X.g <- getis(X, R=0.2)

  plot(X.g,dimyx=c(200,100))

   ## Plot the same, using smooth.ppp in spatstat
    X2 <- setmarks(X, X.g$klocal)

    Z <- Smooth(X2, sigma=0.05, dimxy=256)

    plot(Z, col=topo.colors(128), main="smoothed neighbourhood density")

    contour(Z, add=TRUE)

    points(X, pch=16, cex=0.5)


## End(Not run)
```

---

`gypsophylous` *Spatial point pattern of a plant community*

---

### Description

Locations of plants in a gypsophylous plant community in Central Spain. These are part of the data collected by Romao (2003) that have been analyzed several times (Escudero *et al.* 2005, De la Cruz 2006). The coordinates of the plans are given in cm.

### Usage

```
data(gypsophylous)
```

### Format

An object of class "ppp" of `spatstat` representing the point pattern of plants locations. See `ppp.object` for details of the format.

### Source

Romao, R.L. 2003. *Estructura espacial de comunidades de gipsófitos: interacciones bióticas y constricciones abióticas.* Tesis Doctoral. Universidad Politécnica de Madrid.

### References

De la Cruz, M. 2006. Introducción al análisis de datos mapeados o algunas de las (muchas) cosas que puedo hacer si tengo coordenadas. *Ecosistemas*. 2006/3.

Escudero, A., Romao, R.L., De la Cruz, M. & Maestre, F. 2005. Spatial pattern and neighbour effects on *Helianthemum squamatum* seedlings in a Mediterranean gypsum community. *J. Veg. Sci.*, **16**: 383-390.

### Examples

```
## Not run:

data(gypsophylous)

plot(gypsophylous)


## End(Not run)
```

---

## `haz.ppp` *Easily convert xy data to ppp format*

---

### Description

ppp maker for the impatient layman

### Usage

```
haz.ppp(W)
```

## Arguments

| | |
|---|---|
| `W` | a data frame or matrix with two or three columns (coordinate x, coordinate y, and mark of the point) |

## Details

This naive function easily transform your xy data to the format required by spatstat (version <2.0). It establishes the window of observation as the rectangle defined by the xy range. It asumes that the first two columns are coordinates x and y, and the third (if any) gives the marks of the points.

## Value

A point pattern with the format of spatstat v. < 2.0

## Author(s)

Marcelino de la Cruz

## See Also

`ppp`, `as.ppp`

## Examples

```
## Not run:

data(fig1)

plot(fig1) #typical xyplot

fig1.ppp <- haz.ppp (fig1)

fig1.ppp

plot(fig1.ppp) # point pattern plot of spatstat


## End(Not run)
```

---

`Helianthemum` *Spatial point pattern of Helianthemum squamatum adult plants and seedlings*

---

## Description

Locations of *H. squamatum* adult plants and seedlings in a 6 m x 7 m plot over gypsum soil in Chinchón (near Madrid, Spain). These are part of the data collected by Romao (2003) that have been analyzed several times (Escudero *et al.*2005, De la Cruz 2006, De la Cruz et al. *in press.* ). The coordinates of the plans are given in cm.

## Usage

```
data(Helianthemum)
```

## Format

An object of class "ppp" of `spatstat` representing the point pattern of plants locations marked by their type. See `ppp.object` for details of the format.The dataset has 866 points with the following levels:

**adultHS** adult *H. squamatum* plants

**deadp** dying *H. squamatum* seedlings

**survpl** surviving *H. squamatum* seedlings

## Source

Romao, R.L. 2003. *Estructura espacial de comunidades de gipsófitos: interacciones bióticas y constricciones abióticas.* Tesis Doctoral. Universidad Politécnica de Madrid.

## References

De la Cruz, M. 2006. Introducción al análisis de datos mapeados o algunas de las (muchas) cosas que puedo hacer si tengo coordenadas. *Ecosistemas* 15 (3): 19-39.

De la Cruz, M., Romao, R.L., Escudero, A. and Maestre, F.T. 2008. Where do seedlings go? A spatio-temporal analysis of early mortality in a semiarid specialist. *Ecography*,31 DOI: 10.1111/j.2008.0906-7590.05299 .

Escudero, A., Romao, R.L., De la Cruz, M. & Maestre, F. 2005. Spatial pattern and neighbour effects on *Helianthemum squamatum* seedlings in a Mediterranean gypsum community. *J. Veg. Sci.*, **16**: 383-390.

## Examples

```
data(Helianthemum)
plot(Helianthemum)
```

---

## `ipc.estK` *Fit the (In)homogeneous Poisson Cluster Point Process by Minimum Contrast*

---

## Description

Fits the (In)homogeneous Poisson Cluster point process to a point pattern dataset by the Method of Minimum Contrast.

## Usage

```
ipc.estK(mippp, lambda = NULL, correction = "iso", r = NULL, sigma2 = NULL,
            rho = NULL, q = 1/4, p = 2)


## S3 method for class 'ecespa.minconfit'
plot(x, type="L", add=FALSE, xlim=NULL, ylim=NULL, lwd=c(1,1),
           lty=c(1,2), col=c(1,2), main=NULL, ...)
```

## Arguments

| | |
|---|---|
| `mippp` | Point pattern to which the (I)PCP will be fitted. A point pattern with the `ppp` format of `spatstat`. |
| `lambda` | Optional. Values of the estimated intensity function as a pixel image (object of class "`im`" of spatstat) giving the intensity values at all locations of `mippp`. |
| `correction` | A character item selecting any of the options "border", "bord.modif", "isotropic", "Ripley" or "translate". It specifies the edge correction(s) to be applied in the computation of the $K(r)$ function. |
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions should be evaluated. |
| `sigma2` | Optional. Starting value for the parameter $sigma2$ of the Poisson Cluster process. |
| `rho` | Optional. Starting value for the parameter $rho$ of the Poisson Cluster process. |
| `q` | $q$ exponent of the contrast criterion (see `mincontrast`). |
| `p` | $p$ exponent of the contrast criterion (see `mincontrast`). |
| `x` | An object of class 'ecespa.minconfit', resulting of applying `ipc.estK` to fit a Poisson Cluster Process. |
| `type` | Type of function to be ploted. If type="L", function $L(r)$ [= $sqrt(K(r)/pi) - r$] is ploted. Otherwise, function $K(r)$ is ploted. |
| `add` | Logical. Should the curves be added to another plot? |
| `xlim` | Vector setting the limits of the x-axis. |
| `ylim` | Vector setting the limits of the y-axis. |
| `lwd` | Vector (length=2) setting the line width for ploting the two functions. |
| `lty` | Vector (length=2) setting the line type for ploting the two functions. |
| `col` | Vector (length=2) setting the line color for ploting the two functions. |
| `main` | Optional. Text to appear as a title of the plot. |
| `...` | Additional graphical parameters passed to `link{plot}`. |

## Details

The algorithm fits the (inhomogeneous) Poisson cluster point process (PCP) to a point pattern, by finding the parameters of the (inhomogeneous) Poisson cluster model which give the closest match between the theoretical K function of the Poisson cluster process and the observed K function. For a concise explanation of the PCP see `pc.estK`. For a more detailed explanation of the Method of Minimum Contrast, see `mincontrast` in **spatstat** or Diggle (2003: 86).

The inhomogeneous PCP can be thought of as a thinned process of an homogeneous PCP, where the spatially varying thinning probability $f(s)$ is related to the spatially varying intensity function $lambda(s)$ as $f(s) = lambda(s)/maxlambda(s)$ (Waagepetersen, 2007). As the inhomogeneous K function for the IPCP coincides with the (homogeneous) K function for the corresponding homogeneous PCP, the parameters of the underlying homomgeneous PCP can be estimated as those that give the closest match between the theoretical K function for the homogeneous PCP and the empirical inhomogeneous K function for the observed IPCP.

This Poisson cluster process can be simulated with `rIPCP`.

## Value

`ipc.estK` gives an object of class '`ecespa.minconfit`', basically a list with the following components:

| | |
|---|---|
| `sigma2` | Parameter $sigma^2$. |
| `rho` | Parameter $rho$. |
| `d.theta` | Minimized value of the contrast criterion $D(theta)$. |
| `Kobs` | Values of the observed K-function. |
| `Kfit` | Values of the fitted K-function. |
| `r` | Sequence of distances at which `Kobs` and `Kfit` have been estimated. |
| `data` | Original point pattern. |
| `lambda` | Original intensity function. |
| `dataname` | Name of the original point pattern. |
| `lambdaname` | Name of the original intensity function image. |
| `q` | $q$ exponent of the contrast criterion. |
| `p` | $p$ exponent of the contrast criterion. |

## Author(s)

Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>, inspired by some code of Philip M. Dixon `http://www.public.iastate.edu/~pdixon/`

## References

Diggle, P. J. 2003. *Statistical analysis of spatial point patterns*. Arnold, London.

Waagepetersen, R. P. 2007. An estimating function approach to inference for inhomogeneous Neymann-Scott processes. *Biometrics* 63: 252-258.

## See Also

some functions in **spatstat**: `mincontrast` for a more general implementation of the method of mimimum contrast; `matclust.estK` and `lgcp.estK` fit other appropriate processes for clustered patterns.

## Examples

```
## Not run:

#####################
## Same example as in pc.estK

data(gypsophylous)

## Estimate K function ("Kobs").

gyps.env <- envelope(gypsophylous, Kest, correction="iso", nsim=99)

plot(gyps.env, sqrt(./pi)-r~r,, legend=FALSE)

## Fit Poisson Cluster Process. The limits of integration
## rmin and rmax are setup to 0 and 60, respectively.

cosa.pc2 <- ipc.estK(gypsophylous, r = gyps.env$r[gyps.env$r<=60])

## Add fitted Kclust function to the plot.


plot(cosa.pc2, add=T, lwd=c(3,3))


## A kind of pointwise test of the gypsophylous pattern been a realisation
## of the fitted model, simulating with rIPCP and using function J (Jest).


gyps.env.sim2 <- envelope(gypsophylous, Jest, nsim=99,
                    simulate=expression(rIPCP(cosa.pc2)))


plot(gyps.env.sim2,  main="",legendpos="bottomleft")

#####################
## Inhomogeneous example

data(urkiola)

   #####################
   ## univariate case

   # get univariate pp
   I.ppp <- split.ppp(urkiola)$birch

   # estimate inhomogeneous intensity function
   I.lam <- predict (ppm(I.ppp, ~polynom(x,y,2)), type="trend", ngrid=200)

   # Compute and plot envelopes to Kinhom, simulating from an Inhomogeneous
   #  Poisson Process:
```

```
 I2.env <- envelope( I.ppp,Kinhom, lambda=I.lam, correction="trans",
                            nsim=99, simulate=expression(rpoispp(I.lam)))
plot(I2.env, sqrt(./pi)-r~r, xlab="r (metres)", ylab= "L (r)", col=c(1,3,2,2),legend=FALSE)

 # It seems that there is short scale clustering; let's fit an Inhomogeneous
 # Poisson Cluster Process:

 I.ki <- ipc.estK(mippp=I.ppp, lambda=I.lam, correction="trans")

 # Compute and plot envelopes to Kinhom, simulating from the fitted IPCP:

 Ipc.env <- Ki(I.ki, correction="trans", nsim=99, ngrid=200)

 plot (Ipc.env, xlab="r (metres)", ylab= "L (r)")

 #####################
 ## bivariate case: test independence between birch and quercus in Urkiola

 J.ppp <- split.ppp(urkiola)$oak

 # We want to simulate oak from a homogeneous Poisson model:
 J.ppm <- ppm(J.ppp, trend=~1, interaction=Poisson() )

 IJ.env <- Kci (mod1=I.ki, mod2=J.ppm, nsim=99)

 plot(IJ.env, type=12)

 plot(IJ.env, type=21)


## End(Not run)
```

## K012 *Tests against 'independent labelling'*

### Description

Given a "fixed" point pattern and some process that asign labels (I,J) to another "variable" point pattern, `K012` estimates the combined bivariate K function between the fixed pattern and every type of the variable pattern, and test that they are independent (i.e. that the labels are randomly assigned, irrespectively of the fixed pattern).

### Usage

```
K012(X, fijo, i, j, nsim = 99, nrank = 1, r = NULL,
 correction = "isotropic")
```

## Arguments

| | |
|---|---|
| `X` | Multitype marked point pattern. An object with the ppp format of **spatstat**. |
| `fijo` | Number or character string identifying the mark value of the "fixed" pattern in X |
| `i` | Number or character string identifying the mark value of the I pattern in X |
| `j` | Number or character string identifying the mark value of the J pattern in X |
| `nsim` | Number of simulated point patterns to be generated when computing the envelopes. |
| `nrank` | Integer. Rank of the envelope value amongst the `nsim` simulated values. A rank of 1 means that the minimum and maximum simulated values will be used. |
| `r` | Numeric vector. The values of the argument r at which the K functions should be evaluated. |
| `correction` | A character item selecting any of the options "border", "bord.modif", "isotropic", "Ripley" or "translate". It specifies the edge correction(s) to be applied. |

## Details

This test was developped to answer some questions about the spatial pattern of survival and mortality of seedlings and its relationships with adult plants in a plant community (De la Cruz *et al. In press* ). In order to evaluate the spatial structures of seedlings fates (survive or die), the null hypothesis of random labelling (Cuzick & Edwards 1990, Dixon 2002) would be the appropriate one. This kind of pattern is the result of two hierarchical processes: a first one that generates the pattern of points (seedlings) and other that assign "labels" (i.e. "die", "survive") to the points. On the other hand, to analyze the relationships between the spatial pattern of emerging seedlings and the pattern of adult plants (two patterns that have been generated independently), independence would be the appropriate null hypothesis (Goreaud & Pellisier 2003). However, testing the relationship between the pattern of seedling fates and the pattern of adult plants does not completely fit any of the mentioned hypotheses because, although the pattern of adult plants and the pattern of, e.g., dead seedlings are generated independently, their relationship is conditioned by the dependence of the fate "dead" on the locations of emerging seedlings. This implies that one can not apply the usual technique of toroidal shifting one pattern over the other to test the independence hypothesis. Instead one must permute the label of the focal fate (i.e. survive, die) over the global pattern of seedlings points, keeping the locations and labels of adults fixed. This is the method that `K012` uses to build the envelopes. The bivariate K functions are computed with the Lotwick's and Silverman's (1982) combined estimator (`Kmulti.ls`).

## Value

A list with two elements.

| | |
|---|---|
| `k01` | Bivariate K function of the fixed point pattern and the I variable type, with simulation envelopes |
| `k02` | Bivariate K function of the fixed point pattern and the J variable type, with simulation envelopes |

Each of the above elements is a `fv.object`, essentially a `data.frame` with the following items:

| | |
|---|---|
| `r` | the values of the argument r at which the functions kave been estimated |
| `hi` | upper envelope of simulations |
| `lo` | lower envelope of simulations |

together with the observed corrected estimate of the combined bivariate K function ( `iso`, `trans`, `border`, etc).

## Author(s)

Marcelino de la Cruz <`marcelino.delacruz@upm.es`>

## References

Cuzick, J. and Edwards, R. 1990. Spatial clustering for inhomogeneous populations (with discussion). *Journal of the Royal Statistical Society* B **52**: 73-104.

De la Cruz, M. 2006. Introducción al análisis de datos mapeados o algunas de las (muchas) cosas que puedo hacer si tengo coordenadas. *Ecosistemas* 15 (3): 19-39.

De la Cruz, M., Romao, R.L., Escudero, A. & Maestre, F.T. *In press*. Where do seedlings go? A spatio-temporal analysis of early mortality in a semiarid specialist. *Ecography*.

Dixon, P. M. 2002. Ripley's K function. In *The encyclopedia of environmetrics* (eds. El-Shaarawi, A.H. & Piergorsch, W.W.), pp. 1976-1803. John Wiley & Sons Ltd, NY.

Goreaud, F. and Pelissier, R. 2003. Avoiding misinterpretation of biotic interactions with the intertype K12-function: population independence vs. random labelling hypotheses. *J. Veg. Sci.* **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**: 406-413.

## See Also

`dixon2002` for another segregation test, based in the contingency table of counts of nearest neigbors in a marked point pattern.

## Examples

```
## Not run:

data(Helianthemum)

## Test asociation/repulsion between the fixed pattern of adult
## H. squamatum plants and the "variable" pattern of surviving and
## dead seedlings, with 2.5% and 97.5% envelopes of 999 random
## labellings.


cosa <- K012(Helianthemum, fijo="adultHS", i="deadpl", j="survpl",
             r=seq(0,200,le=201), nsim=999, nrank=25, correction="isotropic")
```

```
plot(cosa$k01, sqrt(./pi)-r~r,  col=c(3, 1, 3), lty=c(3, 1, 3), las=1,
         ylab=expression(L[12]), xlim=c(0, 200),
 main="adult HS vs. dead seedlings")

plot(cosa$k02, sqrt(./pi)-r~r, col=c(3, 1, 3), lty=c(3, 1, 3), las=1,
         ylab=expression(L[12]), xlim=c(0, 200),
   main="adult HS vs. surviving seedlings")

## End(Not run)
```

---

K1K2 *Differences between univariate and bivariate K-functions*

---

## Description

Given two point patterns I and J, `K1K2`computes the differences between both univariate $K$-functions (i.e. $Ki(r) - Kj(r)$) as well as the differences between the univariate and the bivariate $K$-function (i.e. $Ki(r) - Kij(r)$ and $Kj(r) - Kij(r)$). It also computes simulation envelopes to test that that the observed differences are within the range expected asuming the random labelling hypothesis.

## Usage

```
K1K2(X, i, j, nsim = 99, nrank = 1, r = NULL,
 correction = "isotropic")
```

## Arguments

| | |
|---|---|
| `X` | Multitype marked point pattern. An object with the ppp format of **spatstat**. |
| `i` | Number or character string identifying the mark value of the I pattern in X. |
| `j` | Number or character string identifying the mark value of the J pattern in X. |
| `nsim` | Number of simulated point patterns to be generated when computing the envelopes. |
| `nrank` | Integer. Rank of the envelope value amongst the `nsim` simulated values. A rank of 1 means that the minimum and maximum simulated values will be used. |
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions should be evaluated. |
| `correction` | A character item selecting any of the options "border", "bord.modif", "isotropic", "Ripley" or "translate". It specifies the edge correction(s) to be applied. |

## Details

The indiscriminate use of the raw bivariate functions (mainly the $K$ or the $L$-bivariate functions) in ecological studies for testing the association/ repulsion between different point patterns waste some of the most interesting properties of the $K$-function. One of them is that under the random labelling hypothesis every individual pattern would be a random thinning of the corresponding bivariate pattern and therefore $Ki(r) = Kj(r) = Kij(r) = pi * r^2$ (Diggle 2003). Dixon (2002) sugested that some differences of these functions could provide provide interesting ecological information. For example, $Ki(r) - Kj(r)$, has an expected value of 0 for all $r$ distances under random labelling and evaluates the differences in the intensity of aggregation of the two point patterns (e.g., in the example bellow, the pattern of drought and herbivory deaths). Other relevant function is $Ki(r) - Kij(r)$ and the complementary $Kj(r) - Kij(r)$ which evaluate the degree of segregation of every individual pattern, i.e. if every point of the pattern is more -or less- surrounded by other points of the same type than would be expected under the random labelling hypothesis. `K1K2` uses $K^*ij(r)$, the combined estimator of Lotwick and Silverman (a weigthed mean of $Kij(r)$ and $Kji(r)$) as computed by `Kmulti.ls`.

## Value

A list with three elements.

| | |
|---|---|
| `k1k2` | Difference between $Ki(r)$ and $Kj(r)$, with simulation envelopes. |
| `k1k12` | Difference between $Ki(r)$ and $Kij(r)$, with simulation envelopes. |
| `k2k12` | Difference between $Kj(r)$ and $Kij(r)$, with simulation envelopes. |

Each of the above elements is a `fv.object`, essentially a `data.frame` with the following items:

| | |
|---|---|
| `r` | The values of the argument r at which the functions kave been estimated. |
| `hi` | Upper envelope of simulations. |
| `lo` | Lower envelope of simulations. |

together with the observed difference in each case (respectively `K1-K2`, `K1-K12` and `K2-K12`).

## Author(s)

Marcelino de la Cruz <`marcelino.delacruz@upm.es`>

## References

De la Cruz, M. 2006. Introducción al análisis de datos mapeados o algunas de las (muchas) cosas que puedo hacer si tengo coordenadas. *Ecosistemas* 15 (3): 19-39.

De la Cruz, M., Romao, R.L., Escudero, A. and Maestre, F.T. 2008. Where do seedlings go? A spatio-temporal analysis of early mortality in a semiarid specialist. *Ecography*,31 DOI: 10.1111/j.2008.0906-7590.05299 .

Diggle, P.J. 2003. *Statistical analysis of spatial point patterns*. Arnold, London.

Dixon, P. M. 2002. Ripley's K function. In *The encyclopedia of environmetrics* (eds. El-Shaarawi, A.H. & Piergorsch, W.W.), pp. 1976-1803. John Wiley & Sons Ltd, NY.

## Examples

```
## Not run:
data(Helianthemum)

cosa12 <- K1K2(Helianthemum, j="deadpl", i="survpl", r=seq(0,200,le=201),
 nsim=999, nrank=1, correction="isotropic")

## plots of figure 9 in De la Cruz (2006)
plot(cosa12$k1k2, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
         main= "survival- death")

plot(cosa12$k1k12, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
 main="segregation of surviving seedlings")

plot(cosa12$k2k12, lty=c(2, 1, 2), col=c(2, 1, 2), xlim=c(0, 200),
         main= "segregation of dying seedlings")

## End(Not run)
```

---

Kci *Test against non-Poisson (in-)homogeneous models*

---

## Description

Functions to automate testing of 'univariate' and 'bivariate' point pattern hypothesis against non-Poisson (in-)homogeneous models.

## Usage

```
Kci(mod1, mod2, correction="trans", nsim=99, ngrid=200, nrep=1e+05,
     r=NULL, simu="both", spctype=1)

Ki(mod1, correction="trans", nsim=99, ngrid=200, nrep=1e+05, r=NULL,
    spctype=1)

## S3 ploth method for objects of class 'ecespa.kci':
## S3 method for class 'ecespa.kci'
plot(x, type=1, q=0.025, kmean=TRUE, add=FALSE, maine=NULL,
       xlabe=NULL, ylabe=NULL, xlime=NULL, ylime=NULL,
       lty=c(1,2,3), col=c(1,2,3), lwd=c(1,1,1), ...)
```

## Arguments

| | |
|---|---|
| `mod1` | Fitted model. An object of class `ppm` or `ecespa.minconfit`. |
| `mod2` | Fitted model. An object of class `ppm` or `ecespa.minconfit`. |
| `correction` | A character item selecting any of the options "border", "bord.modif", or "translate". It specifies the edge correction to be applied when computing K-functions. |

| | |
|---|---|
| `nsim` | Number of simulated point patterns to be generated when computing the envelopes. |
| `ngrid` | Dimensions (ngrid by ngrid) of a rectangular grid of locations where `predict.ppm` would evaluate the spatial trend of the fitted models. |
| `nrep` | Total number of steps (proposals) of Metropolis-Hastings algorithm that should be run by `rmh` to simulate models of class `ppm`. |
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions should be evaluated. |
| `simu` | A character item indicating if both models will be simulated for the computation of the envelopes (`simu = "both"`) or just the second model (`simu != "both"`). |
| `spctype` | Type of 'pre-thinning' method employed by `rIPCP` in the simulation of `ecespa.minconfit` models. |
| `x` | An object of class 'ecespa.kci'. The result of runing `Kci` or `Ki`. |
| `type` | What to plot. One of 1 (*K1*), 2 (*K2*), 12 (*K12*), 21 (*K21*), 112 (*K1-K12*) or 221 (*K2-K21*). |
| `q` | Quantile for selecting the simulation envelopes. |
| `kmean` | Logical. Should the mean of the simulated envelopes be ploted? |
| `add` | Logical. Should the kci.plot be added to a previous plot? |
| `maine` | Title to add to the plot. |
| `xlabe` | Text or expression to label the x-axis. |
| `ylabe` | Text or expression to label the y-axis. |
| `xlime` | Max and min coordinates for the x-axis. |
| `ylime` | Max and min coordinates for the y-axis. |
| `lty` | Vector with the line type for the estimated Kmm function, the simulated envelopes and the mean of the simulated envelopes. |
| `col` | Vector with the color for the estimated K-function, the simulated envelopes and the mean of the simulated envelopes. |
| `lwd` | Vector with the line width for the estimated K-function, the simulated envelopes and the mean of the simulated envelopes. |
| `...` | Additional graphical parameters passed to plot. |

## Details

These functions are designed to automate the testing of 'univariate' and(/or) 'bivariate' point pattern hypotheses (based on K-functions) against non-Poisson (in-)homogeneous models. These non-Poisson (in-)homogeneous models should have been fitted with pseudolikelihood tools (`spatstat` `ppm` function) or with minimum contrast methods (`ecespa.minconfit`).

Function `Ki` is designed to test 'univariate' hypotheses. It will compute the (in-)homogeneous K-function (using `spatstat` `Kinhom` function) of the point pattern to which the `ppm` or `ecespa.minconfit` model has beeen fitted and will compute 'envelopes' simulating from the fitted model. The computed envelopes can be considered as a pointwise test of the point pattern been a realisation of the fitted model.

Function `Kci` is designed to test 'bivariate' hypotheses. It will compute the (in-)homogeneous cross K-function (using `spatstat` `Kcross.inhom` function) and will compute envelopes simulating from the fitted models. As, when dealing with inhomogeneos patterns $K12$ != $K21$, `Kci` will compute both functions. If `simu = "both"` (default option), `Kci` will simulate from `mod2` to test K12 and from `mod1` to test K21. If `simu != "both"`, only `mod2` will be simulated. This option may be useful when only K12 is of interest. Function `Kci` will also compute univariate (in-) homogeneous K-functions and envelopes for each individual point pattern.

The S3 ploth method will plot the selected K-function and envelopes (actually, it will plot the most usual L-function = $sqrt[K(r)/pi] - r$). The appropriate K function can be selected with the argument `type`. If `type = 1` (default option), it will plot the univariate K function (of the analized model in `Ki` or of the first model [mod1] in `Kci`). If `type = 2`, it will plot the univariate K function of the second model (mod2 in `Kci`). When `type = 12` or `type = 21`, it will plot respectively K12 or K21. Options `type = 112` or `type = 221` will graph a kind of 'segregation test' (see `K1K2`), and will represent de differences K1-K12, K2-K21 and their envelopes.

## Value

Both `Kci` and `Ki` return an object of class `'ecespa.kci'`, basically a list with the following items:

| | |
|---|---|
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions have been evaluated. |
| `kia` | Numeric vector. Observed (in-)homogeneous K function for `mod1` point pattern. |
| `kib` | Numeric vector. Observed (in-)homogeneous K function for `mod2` point pattern. |
| `kci.ab.o` | Numeric vector. Observed (in-) homogeneous cross K-function (K12) for `mod1` and `mod2` point patterns. |
| `kci.ba.o` | Numeric vector. Observed (in-) homogeneous cross K-function (K21) for `mod2` and `mod1` point patterns. |
| `kci.ab.s` | Matrix of simulated (in-) homogeneous cross K-function (K12) for `mod1` and `mod2` point patterns. |
| `kci.ba.s` | Matrix of simulated (in-) homogeneous cross K-function (K21) for `mod2` and `mod1` point patterns. |
| `kib.s` | Matrix of simulated (in-)homogeneous K function for `mod2` point pattern. |
| `kia.s` | Matrix of simulated (in-)homogeneous K function for `mod1` point pattern. |
| `datanamea` | Name of `mod1` point pattern. |
| `datanameb` | Name of `mod2` point pattern. |
| `modnamea` | Name of model `mod1`. |
| `modnameb` | Name of model `mod2`. |
| `type` | Type of analysis. "Kci" or "Ki". |

## Warning

As this implementation involves the use of images as the means of evaluation of the (inhomogeneous) spatial trend, and a mask based on those images will be used as the point pattern window, the "Ripley's" or "isotropic" edge correction can not be employed.

It is usual that during the simulation process some warnings are produced. They are related to some simulated points being rejected as lying outside the specified window.

## Note

Even when one of the two point patterns is assumed to be homogeneous Poisson (and, apparently not worth of fitting any model), an homogeneous Poisson model can be easily fitted and passed to `Kci` with `ppm`. See the examples.

## Author(s)

Marcelino de la Cruz Rot <marcelino.delacruz@upm.es>

## References

De la Cruz, M. and Escudero, A. 2008. Null models and tools for multivariate heterogeneous point patterns. *Submitted*.

De Soto, L., Olano, J.M., Rozas, V. and De la Cruz, M. 2009. Release of *Juniperus thurifera* woodlands from herbivore-mediated arrested succession in Spain. *Applied Vegetation Science*. DOI: 10.1111/j.1654-109X.2009.01045.x .

## Examples

```
## Not run:

   require(spatstat)
   data(urkiola)

 #####################
  ## univariate case

  # get univariate pp
  I.ppp <- split.ppp(urkiola)$birch

  # estimate inhomogeneous intensity function
  I.lam <- predict (ppm(I.ppp, ~polynom(x,y,2)), type="trend", ngrid=200)

  # Compute and plot envelopes to Kinhom, simulating from an Inhomogeneous
  #  Poisson Process:

  I2.env <- envelope( I.ppp,Kinhom, lambda=I.lam, correction="trans",
                             nsim=99, simulate=expression(rpoispp(I.lam)))
 plot(I2.env, sqrt(./pi)-r~r, xlab="r (metres)", ylab= "L (r)", col=c(1,3,2,2),legend=FALSE)

  # It seems that there is short scale clustering; let's fit an Inhomogeneous
  # Poisson Cluster Process:

  I.ki <- ipc.estK(mippp=I.ppp, lambda=I.lam, correction="trans")

  # Compute and plot envelopes to Kinhom, simulating from the fitted IPCP:

  Ipc.env <- Ki(I.ki, correction="trans", nsim=99, ngrid=200)

  plot (Ipc.env, xlab="r (metres)", ylab= "L (r)")
```

```
    #####################
    ## bivariate case: test independence between birch and quercus in Urkiola

    J.ppp <- split.ppp(urkiola)$oak

    # We want to simulate oak from a homogeneous Poisson model:
    J.ppm <- ppm(J.ppp, trend=~1, interaction=Poisson() )

    IJ.env <- Kci (mod1=I.ki, mod2=J.ppm, nsim=99)

    plot(IJ.env, type=12)

    plot(IJ.env, type=21)


## End(Not run)
```

---

## Kinhom.log *Simulation envelopes from the fitted values of a logistic model*

---

### Description

Computes simulation envelopes for (in-)homogeneous K-function simulating from a vector of probabilitiesn.

### Usage

```
Kinhom.log (A, lambda=NULL, mod=NULL, lifemark="0", prob=NULL,
r=NULL, nsim=99, correction="trans", ngrid=200)
```

### Arguments

| | |
|---|---|
| `A` | A marked point pattern with the `ppp` format of `spatstat`. |
| `lambda` | Optional. Values of the estimated intensity function as a pixel image (object of class "`im`" of spatstat) giving the intensity values at all locations of `A`. |
| `mod` | A fitted model. An object of class `ppm`. |
| `lifemark` | Level of the marks of `A` which represents the "live" or "succes" cases. |
| `prob` | Numeric vector, with length equal to the number of points of `A`, represeting the fitted values of a logistic model fitted to `A` marks. |
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions should be evaluated. |
| `nsim` | Number of simulated point patterns to be generated when computing the envelopes. |
| `correction` | A character item selecting any of the options "border", "bord.modif", or "translate". It specifies the edge correction to be applied when computing K-functions. |

| | |
|---|---|
| `ngrid` | Dimensions (ngrid by ngrid) of a rectangular grid of locations where `predict.ppm` would evaluate the spatial trend of the fitted models. |

## Details

This function is a wrapper to compute the critical envelopes for Monte Carlo test of goodness-of-fit of (in-)homogeneous K functions, simulating from the fittted values of a logistic model (i.e. a binomial GLM with logit link) fitted to the marks ("failure", "success") of a "binomially"-marked point pattern. This is particularly interesting in plant ecology when considering alternatives to the *random mortality hypothesis* (Kenkel 1988). This hypothesis is usually tested building Monte Carlo envelopes from the "succesful" patterns resulting from a random labelling of a "binomially"-marked point pattern (this is equivalent to a random thinning of the whole pattern irrespective of the marks). As tree mortality is rarely random but instead can be modelled as a function of a certain number of covariates, the most natural alternative to the *random mortality hypothesis* is the *logistic* mortality hypothesis, that can be tested thinning the original pattern of trees with retention probabilities defined by the fitted values of a logistic model (Batista and Maguire 1998, Olano et al. 2008).

`Kinhom.log` will compute the envelopes by thinning the unmarked point pattern `A` with retention probabilities `prob`. If no `prob` vector is provided, all points will be thinned with the same probability ( number of "live" points / number of points ), i.e. `Kinhom.log` will compute random thinning envelopes.

`Kinhom.log` will compute envelopes both to homogeneous and inhomogeneous K functions. If no `lambda` or `mode` arguments are provided, `Kinhom.log` assumes that the original pattern is homogeneous and will use a constant `lambda` to compute the inhomogeneous K (i.e. it will compute the homogeneous K). The most convenient use with inhomogeneous point patterns is to provide the argument `mod` with an inhomogeneous Poisson model fitted to the original pattern of 'live' points (with spatstat function `ppm`; see the examples). This model will be used to compute (and to update in the simulations) the inhomogeneous trend (i.e. the "lambda") of the patterns. If the argument `lambda` is provided but not `mod`, these lambda will be used as a covariate to fit an inhomogeneous Poisson model that will be used to compute (and to update in the simulations) the inhomogeneous spatial trend.

`Kinhom.log` will produce an object of class 'ecespa.kci' that can be easily ploted (see the examples). This is accomplished by the S3 ploth method `plot.ecespa.kci`; it will plot the K-function and its envelopes (actually, it will plot the most usual L-function = $sqrt[K(r)/pi] - r$).

## Value

`Kinhom.log` returns an object of class `ecespa.kci`, basically a list with the following items:

| | |
|---|---|
| `r` | Numeric vector. The values of the argument $r$ at which the $K(r)$ functions have been evaluated. |
| `kia` | Numeric vector. Observed (in-)homogeneous K function. |
| `kia.s` | Matrix of simulated (in-)homogeneous K functions. |
| `datanamea` | Name of point pattern `A`. |
| `modnamea` | Name of model `mod`. |
| `type` | Type of analysis. Always "Kinhom.log". |
| `probname` | Name of the vector of fitted retention probabilities `prob`. |

| | |
|---|---|
| `modtrend` | Spatial trend (formula) of the model `mod`. |
| `nsim` | Number of simulations. |

## Warning

As this implementation involves the use of images as the means of evaluation of the (inhomogeneous) spatial trend, and a mask based on those images will be used as the point pattern window, the "Ripley's" or "isotropic" edge correction can not be employed.

## Author(s)

Marcelino de la Cruz Rot `<marcelino.delacruz@upm.es>`

## References

Batista, J.L.F. and Maguire, D.A. 1998. Modelling the spatial structure of tropical forests. *For. Ecol. Manag.* 110: 293-314.

Kenkel, N.C. 1988. Pattern of self-thinning in Jack Pine: testing the random mortality hypothesis. *Ecology* 69: 1017-1024.

Olano, J.M., Laskurain, N.A., Escudero, A. and De la Cruz, M. 2009. Why and where adult trees die in a secondary temperate forest? The role of neighbourhood. *Annals of Forest Science* DOI: 10.1051/forest:2008074 .

## Examples

```
 ## Not run:

   require(spatstat)

   data(quercusvm)

   # read fitted values from logistic model:


   probquercus <-c(0.99955463, 0.96563477, 0.97577094, 0.97327199, 0.92437309,
   0.84023396, 0.94926682, 0.89687281, 0.99377915, 0.74157478, 0.95491518,
   0.72366493, 0.66771787, 0.77330148, 0.67569082, 0.9874892, 0.7918891,
   0.73246803, 0.81614635, 0.66446411, 0.80077908, 0.98290508, 0.54641754,
   0.53546689, 0.73273626, 0.7347013, 0.65559655, 0.89481468, 0.63946334,
   0.62101995, 0.78996371, 0.93179582, 0.80160346, 0.82204428, 0.90050059,
   0.83810669, 0.92153079, 0.47872421, 0.24697004, 0.50680935, 0.6297911,
   0.46374812, 0.65672284, 0.87951682, 0.35818237, 0.50932432, 0.92293014,
   0.48580241, 0.49692053, 0.52290553, 0.7317549, 0.32445982, 0.30300865,
   0.73599359, 0.6206056, 0.85777043, 0.65758613, 0.50100406, 0.31340849,
   0.22289286, 0.40002879, 0.29567678, 0.56917817, 0.56866864, 0.27718552,
   0.4910667, 0.47394411, 0.40543788, 0.29571349, 0.30436276, 0.47859015,
   0.31754526, 0.42131675, 0.37468782, 0.73271225, 0.26786274, 0.59506388,
   0.54801851, 0.38983575, 0.64896835, 0.37282031, 0.67624306, 0.29429766,
   0.29197755, 0.2247629, 0.40697843, 0.17022391, 0.26528042, 0.24373722,
   0.26936163, 0.13052254, 0.19958585, 0.18659692, 0.36686678, 0.47263005,
   0.39557661, 0.68048997, 0.74878567, 0.88352322, 0.93851375)
```

```
  ################################
  ## Envelopes for an homogeneous point pattern:

  cosap <- Kinhom.log(A=quercusvm, lifemark="0", nsim=99, prob=probquercus)

  plot(cosap)


  ################################
  ## Envelopes for an inhomogeneous point pattern:

  ## First, fit an inhomogeneous Poisson model to alive trees :

  quercusalive <- unmark(quercusvm[quercusvm$marks == 0])

   mod2 <- ppm(quercusalive, ~polynom(x,y,2))

   ## Now use mod2 to estimate lambda for K.inhom:

   cosapm <- Kinhom.log(A=quercusvm, lifemark="0", prob=probquercus,
                                nsim=99, mod=mod2)


  ################################
  ## An example of homogeneous random thinning:

  cosa <- Kinhom.log(A=quercusvm, lifemark="0")

  plot(cosa)


## End(Not run)
```

---

`Kmm` *Mark-weighted K-function*

---

## Description

This is a functional data summary for marked point patterns that measures the joint pattern of points and marks at different scales determined by $r$.

## Usage

```
Kmm(mippp, r = 1:10, nsim=NULL)

## S3 method for ploting objects of class 'ecespa.kmm':
```

```
## S3 method for class 'ecespa.kmm'
plot(x, type="Kmm.n", q=0.025,
            xlime=NULL, ylime=NULL,  maine=NULL, add=FALSE, kmean=TRUE,
            ylabe=NULL, xlabe=NULL, lty=c(1,2,3), col=c(1,2,3), lwd=c(1,1,1),
             ...)
```

## Arguments

| | |
|---|---|
| mippp | A marked point pattern. An object with the ppp format of **spatstat**. |
| r | Sequence of distances at which Kmm is estimated. |
| nsim | Number of simulated point patterns to be generated when computing the envelopes. |
| x | An object of class 'ecespa.kmm'. The result of applying Kmm to a marked point pattern. |
| type | Type of mark-weighted K-function to plot. One of "Kmm" ("plain" mark-weighted K-function) or "Kmm.n" (normalized mark-weighted K-function). |
| q | Quantile for selecting the simulation envelopes. |
| xlime | Max and min coordinates for the x-axis. |
| ylime | Max and min coordinates for the y-axis. |
| maine | Title to add to the plot. |
| add | Logical. Should the kmm.object be added to a previous plot? |
| kmean | Logical. Should the mean of the simulated Kmm envelopes be ploted? |
| ylabe | Text or expression to label the y-axis. |
| xlabe | Text or expression to label the x-axis. |
| lty | Vector with the line type for the estimated Kmm function, the simulated envelopes and the mean of the simulated envelopes. |
| col | Vector with the color for the estimated Kmm function, the simulated envelopes and the mean of the simulated envelopes. |
| lwd | Vector with the line width for the estimated Kmm function, the simulated envelopes and the mean of the simulated envelopes. |
| ... | Additional graphical parameters passed to plot. |

## Details

Penttinnen (2006) defines $Kmm(r)$, the mark-weighted $K$-function of a stationary marked point process $X$, so that

$$lambda * Kmm(r) = Eo[sum(mo * mn)]/mu^2$$

where $lambda$ is the intensity of the process, i.e. the expected number of points of $X$ per unit area, $Eo[]$ denotes expectation (given that there is a point at the origin); $m0$ and $mn$ are the marks attached to every two points of the process separated by a distance $<= r$ and $mu$ is the mean mark. It measures the joint pattern of marks and points at the scales determmined by $r$. If all the marks are set to 1, then $lambda * Kmm(r)$ equals the expected number of additional random points within a distance $r$ of a typical random point of $X$, i.e. $Kmm$ becomes the conventional Ripley's $K$-function for unmarked point processes. As the $K$-function measures clustering or regularity among the points regardless of the marks, one can separate clustering of marks with the *normalized weighted K-function*

$$Kmm.normalized(r) = Kmm(r)/K(r)$$

If the process is independently marked, $Kmm(r)$ equals $K(r)$ so the normalized mark-weighted $K$-function will equal 1 for all distances $r$.

If `nsim != NULL`, `Kmm` computes *'simulation envelopes'* from the simulated point patterns. These are simulated from `nsim` random permutations of the marks over the points coordinates. This is a kind of pointwise test of $Kmm(r) == 1$ or $normalizedKmm(r) == 1$ for a given $r$.

## Value

`Kmm` returns an object of class `'ecespa.kmm'`, basically a list with the following items:

| | |
|---|---|
| `dataname` | Name of the analyzed point pattern. |
| `r` | Sequence of distances at which `Kmm` is estimated. |
| `nsim` | Number of simulations for computing the envelopes, or `NULL` if none. |
| `kmm` | Mark-weighted $K$-function. |
| `kmm.n` | Normalized mark-weighted $K$-function. |
| `kmmsim` | Matrix of simulated mark-weighted $K$-functions, or or `NULL` if none. |
| `kmmsim.n` | Matrix of simulated normalized mark-weighted $K$-functions, or or `NULL` if none. |

## Note

This implementation estimates $Kmm(r)$ without any correction of border effects, so it must be used with caution. However, as $K(r)$ is also estimed without correction it migth compensate the border effects on the normalized $Kmm$-function.

## Author(s)

Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>

## References

De la Cruz, M. 2008. Métodos para analizar datos puntuales. En: *Introducción al Análisis Espacial de Datos en Ecología y Ciencias Ambientales: Métodos y Aplicaciones* (eds. Maestre, F. T., Escudero, A. y Bonet, A.), pp 76-127. Asociación Española de Ecología Terrestre, Universidad Rey Juan Carlos y Caja de Ahorros del Mediterráneo, Madrid.

Penttinen, A. 2006. Statistics for Marked Point Patterns. In *The Yearbook of the Finnish Statistical Society*, pp. 70-91.

## See Also

`markcorr`

## Examples

```
  ## Not run:
  ## Figure 3.10 of De la Cruz (2008):

  data(seedlings1)

  data(seedlings2)

  s1km <- Kmm(seedlings1, r=1:100)

  s2km <- Kmm(seedlings2, r=1:100)

  plot(s1km, ylime=c(0.6,1.2), lwd=2, maine="", xlabe="r(cm)")

  plot(s2km,  lwd=2, lty=2, add=T )

  abline(h=1, lwd=2, lty=3)

  legend(x=60, y=1.2, legend=c("Hs_C1", "Hs_C2", "H0"),
 lty=c(1, 2, 3), lwd=c(3, 2, 2), bty="n")

## A pointwise test of normalized Kmm == 1 for seedlings1:

   s1km.test <- Kmm(seedlings1, r=1:100, nsim=99)

   plot(s1km.test,  xlabe="r(cm)")


## End(Not run)
```

---

## Kmulti.ls — *Lotwick's and Silverman's combined estimator of the marked K-function*

---

### Description

For a multitype point pattern, calculates the combined estimator of the bivariate $Kij(r)$ and $Kji(r)$ functions.

### Usage

```
Kmulti.ls(X, I, J, r = NULL, corre = "isotropic")
```

### Arguments

| | |
|---|---|
| X | Multitype marked point pattern. An object with the ppp format of **spatstat**. |
| I | Subset index specifying the points of the first pattern. |
| J | Subset index specifying the points of the second pattern. |

| | |
|---|---|
| `r` | Numeric vector. The values of the argument r at which the multitype K function $K^*ij(r)$ should be evaluated. |
| `corre` | A character item selecting any of the options "border", "bord.modif", "isotropic", "Ripley" or "translate", as described in `Kest`. It specifies the edge correction(s) to be applied. |

## Details

As a consequence of edge effects, the estimators $Kij(r)$ and $Kji(r)$ of the same bivariate pattern could differ. $K^*ij(r)$ is the combined estimator defined by Lotwick and Silverman (1982) as

$$nj * Kij(r) + ni * Kji(r)/(ni + nj),$$

$ni$ and $nj$ being respectively the number of points in $I$ and $J$.

## Value

An object of class "fv" (see `fv.object`). Essentially a data frame containing numeric columns

| | |
|---|---|
| `r` | The values of the argument r at which the function $K^*ij(r)$ has been estimated |

.

| | |
|---|---|
| `theo` | The theoretical value of $K * ij(r)$ for a marked Poisson process, namely $pi * r^2$ |

.

together with a column or columns named "border", "bord.modif", "iso" and/or "trans", according to the selected edge corrections. These columns contain estimates of the function $K^*ij(r)$ obtained by the edge corrections named.

## Note

`Kmulti.ls` is a wrapper for a convenient use of the `Kmulti` function of **spatstat**. Please refer to its help page for additional documentation.

## Author(s)

Marcelino de la Cruz. `<marcelino.delacruz@upm.es>`

## References

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**: 406-413.

## Examples

```
## Not run:
data(amacrine)

plot(Kmulti.ls(amacrine, I=amacrine$marks=="on", J=amacrine$marks=="off",
 corre="isotropic"), sqrt(./pi)-r~r, main="")
```

```
# compare with Kmulti

plot(Kmulti(amacrine, I=amacrine$marks=="on", J=amacrine$marks=="off"),
         sqrt(iso/pi)-r~r, add=TRUE, col=3)

plot(Kmulti(amacrine, J=amacrine$marks=="on", I=amacrine$marks=="off"),
      sqrt(iso/pi)-r~r, add=TRUE, col=4)

## End(Not run)
```

## LF.gof *Loosmore and Ford Goodness of Fit Test*

### Description

Performs the Loosmore and Ford (2006) test or the Maximum Absolute Deviation test for a spatial point pattern.

### Usage

```
LF.gof(X, rmin=NULL, rmax=NULL, na.rm=TRUE)
```

### Arguments

| | |
|---|---|
| `X` | An object resulting from the function `envelope`, i.e., with an attribute "*simfuns*" (obtained using the argument `savefuns=TRUE` in `envelope`) ,which is an object of class "`fv`" containing the summary functions computed for each of the simulated patterns. |
| `rmin` | Minimum value of the function argument r over which the maximum absolute deviation, or the integral, will be computed for the test. |
| `rmax` | Maximum value of the function argument r over which the maximum absolute deviation, or the integral, will be computed for the test. |
| `na.rm` | Should NA's be removed to compute the integral? |

### Details

These function perform a tests for goodness-of-fit of a point pattern dataset to a point process model, based on Monte Carlo simulation from the model. The simulations should have been previously computed with the function `envelope`, applied with the argument `savefuns=TRUE` in order to save all the simulated functions, required for the computation of the test.

The test, popularized in the ecological field by Loosmore and Ford (2006) is also described in Diggle (2003, page 14), and according to Baddeley and Turner (2005) also in Diggle (1986) and Cressie (1991, page 667, equation (8.5.42)). If the arguments `rmin` and `rmax` are set to `NULL`, the integral of the GoF statistics will be computed over the complete range of r values.

### Value

A list with the following components:

**u** The GoF statistic, i.e., the value of the integral over the range of *r*'s

**p** The p-value of the test

**na.count.by.r** Number of `NA` values for each r. It helps to evaluate the reliability of the computed u's, specially for small *r*'s

### Author(s)

Marcelino de la Cruz <`marcelino.delacruz@upm.es`>

### References

Cressie, N.A.C. (1991) *Statistics for spatial data*. John Wiley and Sons, 1991.

Diggle, P. J. (1986). Displaced amacrine cells in the retina of a rabbit : analysis of a bivariate spatial point pattern. *J. Neuroscience Methods* 18, 115-125.

Diggle, P.J. (2003) *Statistical analysis of spatial point patterns*, Second edition. Arnold.

Loosmore, N.B. and Ford, E.D. (2006) Statistical inference using the G or K point pattern spatial statistics. *Ecology* 87, 1925-1931.

### See Also

`dclf.test` for an alternative implementation of the test in **spatstat**.

---

## marksum *Mark-sum measure*

---

### Description

An exploratory data analysis technique for marked point patterns. The marked point pattern is mapped to a random field for visual inspection.

### Usage

```
marksum(mippp, R = 10, nx = 30, ny = 30)

## S3 method for ploting objects of class 'ecespa.marksum':
## S3 method for class 'ecespa.marksum'
plot(x, what="normalized",  contour=FALSE, grid=FALSE,
ribbon=TRUE,col=NULL ,main=NULL,xlab="",ylab="",...)
```

## Arguments

| | |
|---|---|
| `mippp` | A marked point pattern. An object with the `ppp` format of **spatstat**. |
| `R` | Radius. The distance argument *r* at which the mark-sum measure should be computed |
| `nx` | Grid density (for estimation) in the x-side. |
| `ny` | Grid density (for estimation) in the y-side. |
| `x` | An object of class `'ecespa.marksum'`. Usually, the result of applying `marksum` to a point pattern. |
| `what` | What to plot. One of `"marksum"` (raw mark sum measure), `"point"` (point sum measure) or `"normalized"` (normalized sum measure). |
| `contour` | Logical; if `"TRUE"` add contour to map. |
| `grid` | Logical; if `"TRUE"` add marked grid to map. |
| `ribbon` | Logical; if `"TRUE"` add legend to map. |
| `col` | Color table to use for the map ( see help file on image for details). |
| `main` | Text or expression to add as a title to the plot. |
| `xlab` | Text or expression to add as a label to axis x. |
| `ylab` | Text or expression to add as a label to axis y. |
| `...` | Additional parameters to `smooth.ppp`, `density.ppp` or `as.mask`, to control the parameters of the smoothing kernel, pixel resolution, etc. |

## Details

Penttinen (2006) defines the *mark-sum measure* as a smoothed summary measuring locally the contribution of points and marks. For any fixed location $x$ within the observational window and a distance $R$, the mark-sum measure $S[R](x)$ equals the sum of the marks of the points within the circle of radius $R$ with centre in $x$. The *point-sum measure* $I[R](x)$ is defined by him as the sum of points within the circle of radius $R$ with centre in $x$, and describes the contribution of points locally near $x$. The *normalized mark-sum measure* describes the contribution of marks near $x$ and is defined (Penttinen, 2006) as

$$S.normalized[R](x) = S[R](x)/I[R](x)$$

This implementation of `marksum` estimates the mark-sum and the point-sum measures in a grid of points whose density is defined by `nx` and `ny`.

## Value

`marksum` gives an object of class `'ecespa.marksum'`; basically a list with the following elements:

| | |
|---|---|
| `normalized` | Normalized mark-sum measure estimated in the grid points. |
| `marksum` | Raw mark-sum measure estimated in the grid points. |
| `pointsum` | Point-sum measure estimated in the grid points. |
| `minus` | Point-sum of the grid points. For advanced use only. |
| `grid` | Grid of points. |

| | |
|---|---|
| `nx` | Density of the estimating grid in the x-side. |
| `ny` | Density of the estimating grid in the x-side. |
| `dataname` | Name of the ppp object analysed. |
| `R` | Radius. The distance argument *r* at which the mark-sum measure has been computed. |
| `window` | Window of the point pattern. |

`plot.ecespa.marksum` plots the selected mark-sum measure.

## Author(s)

Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>

## References

Penttinen, A. 2006. Statistics for Marked Point Patterns. In *The Yearbook of the Finnish Statistical Society*, pp. 70-91.

## See Also

`getis`, related to the point-sum measure, and `markstat` for designing different implementations.

## Examples

```
 ## Not run:

 data(seedlings1)

 seed.m <- marksum(seedlings1, R=25)

 # raw mark-sum measure; sigma is bandwith for smoothing
 plot(seed.m, what="marksum", sigma = 5)

 # point sum measure
 plot(seed.m, what="pointsum", sigma = 5)

 # normalized  mark-sum measure
 plot(seed.m,  what="normalized", dimyx=200, contour=TRUE, sigma = 5)

# the same with added grid and normalized  mark-sum measure
plot(seed.m,  what="normalized", dimyx=200,
      contour=TRUE, sigma = 5, grid=TRUE)


## End(Not run)
```

`p2colasr` *P-value for a discrete distribution on small sample data*

## Description

Computes the p-value for a two-sided hypothesis test following Dixon's (2002:145) description of the method of Agresti & Min (2001).

## Usage

```
p2colasr(Z, nsim = length(Z))
```

## Arguments

| | |
|---|---|
| `Z` | `vector` with the observed Z-score in the first position and all the simulated values behind. |
| `nsim` | Number of simulated values. |

## Value

P-value of the two-sided hypothesis test

## Note

This function is usually not to be called by the user. It is internally used by `dixon2002`.

## Author(s)

Marcelino de la Cruz Rot. `<marcelino.delacruz@upm.es>`

## References

Agresti, A. & Min, Y. 2001. On small-sample confidence intervals for parameters in discrete distributions. *Biometrics*, **57**: 963-971.

Dixon, P.M. 2002. Nearest-neighbor contingency table analysis of spatial segregation for several species. *Ecoscience*, **9**(2): 142-151.

## pc.estK — *Fit the Poisson Cluster Point Process by Minimum Contrast*

### Description

Fits the Poisson Cluster point process to a point pattern dataset by the Method of Minimum Contrast.

### Usage

```
pc.estK(Kobs, r, sigma2 = NULL, rho = NULL)
Kclust(r, sigma2, rho)
```

### Arguments

| | |
|---|---|
| `Kobs` | Empirical $K$-function. |
| `r` | Sequence of distances at which function $K$ has been estimated. |
| `sigma2` | Optional. Starting value for the parameter $sigma2$ of the Poisson Cluster process. |
| `rho` | Optional. Starting value for the parameter $rho$ of the Poisson Cluster process. |

### Details

The algorithm fits the Poisson cluster point process to a point pattern, by finding the parameters of the Poisson cluster model which give the closest match between the theoretical K function of the Poisson cluster process and the observed K function. For a more detailed explanation of the Method of Minimum Contrast, see `mincontrast` in **spatstat** or Diggle (2003: 86).

The Poisson cluster processes are defined by the following postulates (Diggle 2003):

*PCP1* Parent events form a Poisson process with intensity $rho$.

*PCP2* Each parent produces a random number of offspring, according to a probability distribution $p[s] : s = 0, 1, 2, ...$

*PCP3* The positions of the offspring relative to their parents are distributed according to a bivariate pdf $h$.

This implementation asumes that the probability distribution $p[s]$ of offspring per parent is a Poisson distribution and that the position of each offspring relative to its parent follows a radially symetric Gaussian distribution with pdf

$$h(x, y) = [1/(2 * pi * sigma^2)] * exp[-(x^2 + y^2)/(2 * sigma^2)]$$

The theoretical $K$-function of this Poisson cluster process is (Diggle, 2003):

$$pi * r^2 + [1 - exp(-r^2/4 * sigma^2)]/rho$$

The command `Kclust` computes the theoretical $K$-function of this Poisson cluster process and can be used to find some initial estimates of $rho$ and $sigma^2$. In any case, the optimization usually finds the correct parameters even without starting values for these parameters.

This Poisson cluster process can be simulated with `sim.poissonc`.

## Value

| | |
|---|---|
| `sigma2` | Parameter $sigma^2$. |
| `rho` | Parameter $rho$. |

## Note

The exponents $p$ and $q$ of the contrast criterion (see `mincontrast`) are fixed respectively to $p = 2$ and $q = 1/4$. The $rmin$ and $rmax$ limits of integration of the contrast criterion are set up by the sequence of values of $r$ and $Kobs$ passed to `pc.estK`.

## Author(s)

Marcelino de la Cruz Rot <marcelino.delacruz@upm.es>, inspired by some code of Philip M. Dixon http://www.public.iastate.edu/~pdixon/

## References

Diggle, P. J. 2003. *Statistical analysis of spatial point patterns*. Arnold, London.

## See Also

`ipc.estK` for fitting the inhomogeneous Poisson cluster process; some functions in **spatstat** ( `matclust.estK` and `lgcp.estK`) fit other appropriate processes for clustered patterns; `mincontrast` performs a more general implementation of the method of mimimum contrast.

## Examples

```
## Not run:

data(gypsophylous)

## Estimate K function ("Kobs").

gyps.env <- envelope(gypsophylous, Kest, correction="iso", nsim=99)

plot(gyps.env, sqrt(./pi)-r~r, legend=FALSE)

## Fit Poisson Cluster Process. The limits of integration
## rmin and rmax are setup to 0 and 60, respectively.

cosa.pc <- pc.estK(Kobs = gyps.env$obs[gyps.env$r<=60],
           r = gyps.env$r[gyps.env$r<=60])

## Add fitted Kclust function to the plot.

lines(gyps.env$r,sqrt(Kclust(gyps.env$r, cosa.pc$sigma2,cosa.pc$rho)/pi)-gyps.env$r,
       lty=2, lwd=3, col="purple")
```

```
## A kind of pointwise test of the gypsophylous pattern been a realisation
## of the fitted model, simulating with sim.poissonc and using function J (Jest).

gyps.env.sim <- envelope(gypsophylous, Jest, nsim=99,
                    simulate=expression(sim.poissonc(gypsophylous,
    sigma=sqrt(cosa.pc$sigma2), rho=cosa.pc$rho)))

plot(gyps.env.sim,  main="",legendpos="bottomleft")


## End(Not run)
```

## quercusvm *Alive and dead oak trees*

### Description

Locations of alive and dead oak trees (*Quercus robur*) in a secondary wood in Urkiola Natural Park (Basque country, north of Spain). This is part of a more extensive dataset collected and analysed by Laskurain (2008). The coordinates of the trees are given in meters.

### Usage

```
data(quercusvm)
```

### Format

An object of class "ppp" representing the point pattern of tree locations. Entries include

**x** Cartesian x-coordinate of tree.

**y** Cartesian y-coordinate of tree.

**marks** factor with two levels indicating the status of each tree (1 = "alive", 0 = "dead").

See ppp for details of the format of a ppp object.

### References

Laskurain, N. A. (2008) *Dinámica espacio-temporal de un bosque secundario en el Parque Natural de Urkiola (Bizkaia).* Tesis Doctoral. Universidad del País Vasco /Euskal Herriko Unibertsitatea.

## rIPCP *Simulate Inhomogeneous Poisson Cluster Process*

### Description

Generate a random point pattern, a simulated realisation of the Inhomogeneous Poisson Cluster Process.

### Usage

```
rIPCP(x, lambda = NULL, type = 1, lmax = NULL, win = owin(c(0, 1), c(0, 1)), ...)
```

### Arguments

| | |
|---|---|
| `x` | an object of class `'ecespa.minconfit'`, resulting from the function `ipc.estK`. |
| `lambda` | Optional. Values of the estimated intensity function as a pixel image (object of class "im" of `spatstat`) giving the intensity values at all locations. |
| `type` | Type of 'prethining' employed in the simulation. See details. |
| `lmax` | Optional. Upper bound on the values of lambda. |
| `win` | Optional. Window of the simulated pattern. |
| `...` | Optional. Arguments passed to `as.im`. |

### Details

This function simulates the Inhomogeneous Poisson Cluster process from an object of class `'ecespa.minconfit'`, resulting from fitting an IPCP to some 'original' point pattern using the function `ipc.estK`. Following the approach of Waagepetersen (2007), the simulation involves a first step in which an homogeneous aggregated pattern is simulated (from the fitted parameters of the `'ecespa.minconfit'` object, using function `rThomas` of `spatstat`) and a second one in which the homogeneous pattern is thinned with a spatially varying thinning probability *f (s)* proportional to the spatially varying intensity, i.e. *f (s) = lambda(s) / max[lambda(s)]*. To obtain a 'final' density similar to that of the original point pattern, a "prethinning" must be performed. There are two alternatives. If the argument `'type'` is set equal to '1', the expected number of points per cluster (*mu* parameter of `rThomas` is thinned as *mu <- mu.0 / mean[f(s)]*, where *mu.0* is the mean number of points per cluster of the original pattern. This alternative produces point patterns most similar to the 'original'. If the argument `'type'` is set equal to '2', the fitted intensity of the Poisson process of cluster centres (*kappa* parameter of `rThomas`, i.e. the intensity of 'parent' points) is thinned as *kappa <- kappa / mean[f(s)]*. This alternative produces patterns more uniform than the 'original' and it is provided only for experimental purposes.

### Value

A point pattern, with the format of the `ppp` objects of `spatstat`.

### Author(s)

Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`>

## References

Waagepetersen, R. 2007. An estimating function approach to inference for inhomogeneous Neyman-Scott processes. *Biometrics* 63:252-258.

## See Also

`sim.poissonc` to simulate homogeneous PCP; `rNeymanScott` and `rThomas` in **spatstat** are the basis of this function

## Examples

```
## Not run: 

  data(gypsophylous)

  plot(gypsophylous)

  ## It 'seems' that the pattern is clustered, so
  ## fit a Poisson Cluster Process. The limits of integration
  ## rmin and rmax are setup to 0 and 60, respectively.

 cosa.pc2 <- ipc.estK(gypsophylous, r = seq(0, 60, by=0.2))

 ## Create one instance of the fitted PCP:

 pointp <- rIPCP( cosa.pc2)

 plot(pointp)



  #####################
  ## Inhomogeneous example

  data(urkiola)

  # get univariate pp
  I.ppp <- split.ppp(urkiola)$birch

  plot(I.ppp)

  #estimate inhomogeneous intensity function
  I.lam <- predict (ppm(I.ppp, ~polynom(x,y,2)), type="trend", ngrid=200)

  # It seems that there is short scale clustering; lets fit an IPCP:

  I.ki <- ipc.estK(mippp=I.ppp, lambda=I.lam, correction="trans")

  ## Create one instance of the fitted PCP:

  pointpi <- rIPCP( I.ki)
```

```
    plot(pointpi)

## End(Not run)
```

---

`seedlings` *Cohorts of Helianthemum squamatum seedlings*

---

### Description

Marked point patterns of two consecutive cohorts of seedlings of *H. squamatum* growing in a gypsophylous plant community in Central Spain. The datasets contains the locations of the seedlings marked with their heigth. Both the coordinates and the heigth of the seedlings are given in cm.

### Usage

```
data(seedlings1)
data(seedlings2)
```

### Format

`seedlings1` and `seedlings2` are objects of class "ppp" representing the point pattern of seedling locations marked by their heights. See `ppp.object` for details of the format.

### Source

Romao, R.L. 2003. *Estructura espacial de comunidades de gipsófitos: interacciones bióticas y constricciones abióticas.* Tesis Doctoral. Universidad Politécnica de Madrid.

### References

De la Cruz, M. 2006. Introducción al análisis de datos mapeados o algunas de las (muchas) cosas que puedo hacer si tengo coordenadas. *Ecosistemas*. 2006/3.

Escudero, A., Romao, R.L., De la Cruz, M. & Maestre, F. 2005. Spatial pattern and neighbour effects on *Helianthemum squamatum* seedlings in a Mediterranean gypsum community. *J. Veg. Sci.*, **16**: 383-390.

### Examples

```
## Not run:

 data(seedlings1)

 plot(seedlings1)


## End(Not run)
```

## `sim.poissonc` *Simulate Poisson Cluster Process*

### Description

Generate a random point pattern, a simulated realisation of the Poisson Cluster Process

### Usage

```
sim.poissonc(x.ppp, rho, sigma)
```

### Arguments

| | |
|---|---|
| `x.ppp` | Point pattern whose window and intensity will be simulated. An object with the `ppp` format of **spatstat**. |
| `rho` | Parameter $rho$ of the Poisson Cluster process. |
| `sigma` | Parameter $sigma$ of the Poisson Cluster process. |

### Details

The Poisson cluster processes are defined by the following postulates (Diggle 2003):

| | |
|---|---|
| *PCP1* | Parent events form a Poisson process with intensity $rho$. |
| *PCP2* | Each parent produces a random number of offspring, according to a probability distribution $p[s] : s = 0, 1, 2, ...$ |
| *PCP3* | The positions of the offspring relative to their parents are distributed according to a bivariate pdf $h$. |

This implementation asumes that the probability distribution $p[s]$ of offspring per parent is a Poisson distribution and that the position of each offspring relative to its parent follows a radially symetric Gaussian distribution with pdf

$$h(x, y) = [1/(2 * pi * sigma^2)] * exp[-(x^2 + y^2)/(2 * sigma^2)]$$

### Value

The simulated point pattern (an object of class "ppp").

### Warning

This implementation simulates only point patterns within rectangular windows. Use `ipc.estK` to fit and `rIPCP` (or the `spatstat` functions) to simulate point patterns within irregular windows.

### Note

This function can use the results of `pc.estK` to simulate point patterns from a fitted model. Be careful as the paramted returned by `pc.estK` is $sigma^2$ while `sim.poissonc` takes its square root, i.e. $sigma$.

## Author(s)

Marcelino de la Cruz Rot <marcelino.delacruz@upm.es>

## References

Diggle, P.J. 2003. *Statistical analysis of spatial point patterns.* Arnold, London.

## See Also

rIPCP to simulate inhomogeneous PCP; rNeymanScott and rThomas in **spatstat**

## Examples

```
## Not run:

data(gypsophylous)

## Estimate K function ("Kobs").
gyps.env <- envelope(gypsophylous, Kest, correction="iso")

plot(gyps.env, sqrt(./pi)-r~r)

## Fit Poisson Cluster Process. The limits of integration
## rmin and rmax are setup to 0 and 60, respectively.
cosa.pc <- pc.estK(Kobs = gyps.env$obs[gyps.env$r<=60],
               r = gyps.env$r[gyps.env$r<=60])

## Add fitted Kclust function to the plot.
lines(gyps.env$r,sqrt(Kclust(gyps.env$r, cosa.pc$sigma2,cosa.pc$rho)/pi)-gyps.env$r,
       lty=2, lwd=3, col="purple")

## A kind of pointwise test of the pattern gypsophilous been a realisation
## of the fitted model, simulating with sim.poissonc and using function J (Jest).

gyps.env.sim <- envelope(gypsophylous, Jest,
                    simulate=expression(sim.poissonc(gypsophylous,
    sigma=sqrt(cosa.pc$sigma2), rho=cosa.pc$rho)))

plot(gyps.env.sim,  main="")


## End(Not run)
```

## `swamp` *Tree Species in a Swamp Forest*

### Description

Locations and botanical classification of trees in a plot in the Savannah River. Locations are given in metres, rounded to the nearest 0.1 metre. The data come from a 1-ha (200 m x 50 m) plot in the Savannah River Site, South Carolina, USA. The 734 mapped stems included 156 Carolina ash (*Fraxinus caroliniana*), 215 Water tupelo (*Nyssa aquatica*), 205 Swamp tupelo (*Nyssa sylvatica*), 98 Bald cypress (*Taxodium distichum*) and 60 stems of 8 additional species. Although the plots were set up by Bill Good and their spatial patterns described in Good and Whipple(1982), the plots have been maintained and resampled by Rebecca Sharitz and her colleagues of the Savannah River Ecology Laboratory. There are slightly different versions of the Good plot data. Every time the plots are resampled, some errors are corrected. This is mostly a concern for the biologists. The different versions are very similar; they are all very good examples of a marked spatial point pattern.

### Usage

```
data(swamp)
```

### Format

A data frame with 734 observations on the following 3 variables.

`x` Cartesian x-coordinate of tree

`y` Cartesian y-coordinate of tree

`sp` a factor with levels indicating the species of each tree:

| | |
|---|---|
| `FX` | Carolina ash (*Fraxinus caroliniana*) |
| `NS` | Swamp tupelo (*Nyssa sylvatica*) |
| `NX` | Water tupelo (*Nyssa aquatica*) |
| `TD` | Bald cypress (*Taxodium distichum*) |
| `OT` | Other species |

### Source

Philip Dixon's personal web page `http://www.public.iastate.edu/~pdixon/`

### References

Dixon, P.M. 2002. Nearest-neighbor contingency table analysis of spatial segregation for several species. *Ecoscience*, **9**(2): 142-151.

Good, , B. J. & Whipple, S.A. 1982. Tree spatial patterns: South Carolina bottomland and swamp forest. *Bulletin of the Torrey Botanical Club*, **109**: 529-536.

Jones et al. 1994. Tree population dynamics in seven South Carolina mixed-species forests. *Bulletin of the Torrey Botanical Club*, **121**:360-368.

### Examples

```
data(swamp)
plot(swamp$x,swamp$y, col=as.numeric(swamp$sp),pch=19,
 xlab="",ylab="",main="Swamp forest")
```

---

## syrjala — *Syrjala's test for the difference between the spatial distributions of two populations*

---

### Description

Computes a two-sample Cramer-von Mises (and Kolmogorov-Smirnov) type test for a difference between the spatial distributions of two populations. It is designed to be sensitive to differences in the way the populations are distributed across the study area but insensitive to differences in abundance between the two populations.

### Usage

```
syrjala0(coords, var1, var2, nsim, R=FALSE)
syrjala(coords = NULL, var1 = NULL, var2 = NULL, nperm = 999)
syrjala.test(ppp1, ppp2, nsim = 999)
## S3 method for class 'syrjala.test'
plot(x, coline=1, ...)
## S3 method for class 'ecespa.syrjala'
plot(x, ...)
```

### Arguments

| | |
|---|---|
| `coords` | A `data.frame` with '$x' and '$y' components. |
| `var1` | The first numeric variable |
| `var2` | The second numeric variable. |
| `nperm` | Number of permutations. |
| `nsim` | Number of permutations. |
| `R` | Logical. Should be computed using R approach? |
| `ppp1` | A marked point pattern, with the `ppp` format of spatstat, representing the values of some parameter measured on the corresponding sampling locations. |
| `ppp2` | A marked point pattern, with the `ppp` format of spatstat, representing the values of some other parameter measured on the same locations than `ppp1`. |
| `x` | An object of class '`syrjala.test`' or '`ecespa.syrjala`' resulting from `syrjala` or `syrjala.test`, respectively. |
| `coline` | color for drawing the statistic's line in the plot. |
| `...` | Graphical parameters passed to `hist`. |

## Details

The null hypothesis of Syrjala's test is that across the study area, the normalized distributions of the two populations are the same (Syrjala, 1996). Population density data are collected at $K$ sampling locations on two populations. Let $(xk, yk)$ denote the coordinates of the $kth$ sampling location ($k = 1, ..., K$ ); let $d.i(xk, yk)$ denote the sample density at the $Kth$ sampling location of the $ith$ population. To construct a test that is independent of the population sizes, the observed density data is first normalized:

$$gamma.i(xk, yk) = di(xk, yk)/Di,$$

where $Di$ is the sum of $d.i(xk, yk)$ observations across the $K$ sampling locations. The value of the cumulative distribution function at the location $(xk, yk)$ for the $ith$ population, denoted $GAMMA.i(xk, yk)$, is the sum of all normalized density observations, $gamma.i(xk, yk)$, whose location $(x, y)$ is such that $x <= xk$ and $y <= yk$. The statistic proposed by Syrjala to test the null hypothesis is the square of the difference between the cumulative distribution functions of the two populations, summed over all sampling locations, that is

$$psi = sumGAMMA.1(xk, yk) - GAMMA.2(xk, yk)^2.$$

As $psi$ is not invariant with respect to the 'corner' of the rectangle enclosing the study area that is chosen as the origin of the coordinate sytem, $psi$ is computed four times, one with each corner as the origin, and the average $psi$ is employed as the test statistic. The level of significance of the observed $psi$ is determined from its position in the ordered set of test statistic values from all $2^K$ pairwise permutations (that is approximated from a large number of randomly selected permutations).

## Value

Functions `syrjala` or `syrjala0` (with the argument `R=FALSE`) return an object of class '`syrjala.test`'. Functions `syrjala.test` or `syrjala0` (with the argument `R=TRUE`) return an object of class '`ecespa.syrjala`'. In Both cases, the result is a list with the following elements:

| | |
|---|---|
| `cvm.obs` | (class syrjala.test). The observed (averaged) $psi$ statistic for the CvM test. |
| `cvm.sim` | (class syrjala.test). A numeric vector with the `nperm+1` simulated $psi$'s statistics (including `cvm.obs`). |
| `ks.obs` | (class syrjala.test). The observed (averaged) $psi$ statistic for the K-S test. |
| `ks.sim` | (class syrjala.test). A numeric vector with the `nperm+1` simulated $psi$'s statistics (including `ks.obs`). |
| `datanames` | (class syrjala.test). A character vector with the names of the two patterns, the spatial congruence of which is been analyzed. |
| `nperm` | (class syrjala.test). The number of permutations employed in the test (not counting the original data). |
| `psi.obs` | (class ecespa.syrjala).The observed (averaged) $psi$ statistic. |
| `psi.sim` | (class ecespa.syrjala). A vector with the `nsim` simulated $psi$'s statistics. |
| `datanames` | (class ecespa.syrjala). A vector with the names of the two point patterns whose spatial congruence is been analyzed. |
| `nsim` | (class ecespa.syrjala). The number of permutations employed in the test. |

Both S3 plot methods plot an histogram with the distribution of the simulated $psi$'s statistics and draws the observed $psi$ as a vertical line.

## Warning

The test requires both populations being sampled in exactly the same sampling locations. Althoug this implementation employs `ppp`'s as the supporting data format, this kind of data are **not** spatial point patterns. They cannot be analysed with the usual tools employed for marked point patterns.

## Note

`syrjala` or `syrjala0` (with the argument `R=FALSE`) implement a Fortran version of Syrjala's test. They run considerably faster than the "whole-R" implementation of `syrjala.test` or `syrjala0` (with the argument `R=TRUE`). This last implementation is supplied for illustrative purposes and to maintain compability with previous versions of package `ecespa`. One can use function `haz.ppp` to easily build the ppp objects from a `data.frame` with only three columns (x-coordinate, y-coordinate and abundance).

This function has been employed to compute Syrjala's test in Rey-Benayas et al. (2008).

## Author(s)

Jose M. Blanco-Moreno <`jmblanco@ub.edu`> for the Fortran implementation of Syrjala's original QBasic function, Marcelino de la Cruz Rot <`marcelino.delacruz@upm.es`> for the R version, the wrapping functions and the documentation

## References

Rey-Benayas, J.M., de la Montaña, E., Pérez-Camacho, L., de la Cruz, M., Moreno, D., Parejo, J.L. and Suárez-Seoane, S. 2008. Inter-annual dynamics and spatial congruence of a nocturnal bird assemblage inhabiting a Mediterranean agricultural mosaic. *Submitted.*

Syrjala, S. E. 1996. A statistical test for a difference between the spatial distributions of two populations. *Ecology* 77: 75-80.

## Examples

```
## Not run:

   data(syr1); data(syr2); data(syr3)

   plot(syrjala.test(syr1, syr2, nsim=999))

   plot(syrjala.test(syr1, syr3, nsim=999))


   coords <- data.frame(x=syr1$x, y=syr1$y); var1<- syr1$marks; var2<- syr2$marks

   syrjala(coords, var1, var2, 9999)

   syrjala0(coords, var1, var2, 9999)

   syrjala0(coords, var1, var2, 999, R=TRUE)


   coords <- expand.grid(x=1:10,y=1:10)
```

```
    var1 <- runif(100)
    var2 <- runif(100)
    syrjala(coords, var1, var2, 9999)



## End(Not run)
```

---

`syrjala.data` *Syrjala test data*

---

### Description

Artificial data to exemplify Syrjala's test.

### Usage

```
data(syr1)
data(syr2)
data(syr3)
```

### Format

`syr1`, `syr2` and `syr3` are marked point patterns of class "ppp" representing the coordinates of some sampling locations, 'marked' by the value of some parameters (e.g. density of individuals) measured on them. See `ppp.object` for details of the format. On the other hand, one can use function `haz.ppp` to easily build ppp objects appropriate for use with `syrjala.test`.

### Examples

```
data(syr1)
```

# Index