Package ‘onpoint’

Type Package
Title Helper Functions for Point Pattern Analysis
Version 1.0.4
Maintainer Maximilian H.K. Hesselbarth <mhk.hesselbarth@gmail.com>
Description Growing collection of helper functions for point pattern analysis. Most functions are designed to work with the 'spatstat' (<http://spatstat.org>) package. The focus of most functions are either null models or summary functions for spatial point patterns. For a detailed description of all null models and summary functions, see Wiegand and Moloney (2014, ISBN:9781420082548).
URL https://r-spatialecology.github.io/onpoint/
BugReports https://github.com/r-spatialecology/onpoint/issues
License GPL (>= 3)
Depends R (>= 3.1)
Imports ggplot2, spatstat.explore, spatstat.geom, spatstat.random, stats
Suggests covr, spatstat (>= 2.0.0), testthat
Encoding UTF-8
RoxygenNote 7.2.3
NeedsCompilation no
Author Maximilian H.K. Hesselbarth [aut, cre]
(https://orcid.org/0000-0003-1125-9918)
Repository CRAN
Date/Publication 2023-03-07 15:10:02 UTC

R topics documented:

  balance_points .................................................. 2
  center_l_function .............................................. 3
  estimate_o_ring .................................................. 4
  estimate_pcf_fast .............................................. 5
balance_points

Description
Balance number of points

Usage
balance_points(pattern, n, verbose = TRUE)

Arguments
pattern     ppp object.
n          Either an integer or a ppp object.
verbose     Print messages.

Details
The function balances out the number of points in the input pattern to either the provided number of points as integer or the same number of points if a ppp object is provided.

Value
ppp

Examples
set.seed(42)
input <- spatstat.random::rpoispp(lambda = 100)
input_b <- spatstat.random::rpoispp(lambda = 100)

balance_points(pattern = input, n = 110)
balance_points(pattern = input, n = input_b)
Description

Centered L-function

Usage

center_l_function(x, 

Arguments

x ppp

... Arguments passed to spatstat.explore::Lest()

Details

Centers Besag’s L-function to zero by calculating $L(r) - r$. Centering the L-function allows an easier interpretation and plotting of the results (Haase 1995).

Returns an ‘Function value object’ of the spatstat package.

Value

fv.object

References


See Also

Lest

Examples

input_pattern <- spatstat.random::runifpoint(n = 100)
center_l_function(input_pattern, correction = "Ripley")

lest <- spatstat.explore::Lest(input_pattern)
center_l_function(lest)
Description

O-ring function

Usage

estimate_o_ring(x, ...)

Arguments

x ppp
...
Arguments passed to spatstat.explore::pcf.ppp()

Details

Estimates the O-ring function proposed by Wiegand and Moloney (2004). The O-ring statistic is defined as:

\[ O(r) = \lambda * g(r) \]

Generally speaking, \( O(r) \) scales the pair correlation \( g(r) \) function with help of the intensity \( \lambda \). One advantage of the O-ring statistic is that it can be interpreted as a neighborhood density because it is a probability density function (Wiegand & Moloney 2004, 2014).

Returns an 'Function value object' of the spatstat package.

Value

fv.object

References


See Also

density.ppp
pcf
**estimate_pcf_fast**

**Examples**

```r
input_pattern <- spatstat.random::runifpoint(n = 100)
estimate_o_ring(input_pattern)
```

**Description**

Fast estimation of the pair correlation function

**Usage**

```r
estimate_pcf_fast(pattern, ...)
```

**Arguments**

- `pattern` Point pattern.
- `...` Arguments passed down to `Kest` or `pcf.fv`.

**Details**

The functions estimates the pair correlation functions based on an estimation of Ripley’s K-function. This makes it computationally faster than estimating the pair correlation function directly.

It is a wrapper around `Kest` and `pcf.fv` and returns a 'Function value object' of the `spatstat` package.

**Value**

`fv.object`

**References**


**See Also**

- `Kest`
- `pcf.fv`
Examples
set.seed(42)
pattern <- spatstat.random::runifpoint(n = 100)
pcf_fast <- estimate_pcf_fast(pattern)

Description
Plotting method for env_summarized object

Usage
## S3 method for class 'env_summarized'
plot(
  x,
  col = c("#97CBDE", "#E1B0B5"),
  x_lab = NULL,
  y_lab = NULL,
  base_size = 10,
  label = TRUE,
  ...
)

Arguments
  
  x     Random patterns.
  col   Colors for areas above and below envelope.
  x_lab, y_lab  Labels of x- and y-axis.
  base_size  Base size of plot
  label     If TRUE the ratios of the area above and below are added to the plot.
  ...      To be generic for plotting function.

Details
Plotting method for summarized envelope created with summarize_envelope.
Returns a ggplot object.

Value
ggplot
### plot_quantums

#### Description

Plot simulation envelopes

#### Usage

```r
plot_quantums(
  input, 
  labels = NULL, 
  color_scale = NULL, 
  legend_position = "bottom", 
  quantum_position = NULL, 
  title = NULL, 
  xlab = NULL, 
  ylab = NULL, 
  line_size = 0.5, 
  base_size = 15, 
  full_fun = TRUE, 
  quantum = TRUE, 
  standarized = FALSE
)
```

#### Arguments

- **input**
  - envelope.
- **labels**
  - Name of the labels. See details for more information.
- **color_scale**
  - Colors used with labels.

#### Examples

```r
set.seed(42)
input_pattern <- spatstat.random::rThomas(kappa = 15, scale = 0.05, mu = 5)

cluster_env <- spatstat.explore::envelope(input_pattern, fun = "pcf", nsim = 39, 
funargs = list(divisor = "d", correction = "Ripley", stoyan = 0.25))
x <- summarize_envelope(cluster_env)
plot(x)
```
plot_quantums

legend_position
The position of legends ("none", "left", "right", "bottom", "top", or two-element numeric vector)

quantum_position
Vector with minimum and maximum y value of the quantum bar.

title
Plot title.

xlab, ylab
axis labels.

line_size
Size of the lines.

base_size
Base font size.

full_fun
If true observed value and envelope is plotted.

quantum
If true quantums bars are plotted.

standarized
If true observed value is standarized. See details for more details.

Details
This functions provides a plotting style for envelope objects of the spatstat package (for more information please see spatstat.explore::envelope). The location of the observed value in relation to the simulation envelope of the null model input is indicated by an additional colour bar at the bottom of the plot. If standarized = TRUE, all values are standarized by subtracting the theoretical value for CSR.

Labels must be a vector including labels for the following three cases. The color scale vector is used in the same order.
1 = observed > high
2 = low < observed < high
3 = observed < low

To adjust the position of the quantum bar, use quantum_position.

Returns a ggplot object.

Value
ggplot

References

See Also
evelope

Examples
set.seed(42)
pattern <- spatstat.random::rThomas(kappa = 50, scale = 0.025, mu = 5)
csr_envelope <- spatstat.explore::envelope(pattern, fun = spatstat.explore::pcf, nsim = 19)
plot_quantums(csr_envelope, ylab = "g(r)")
print.env_summarized

Description

Print method for env_summarized object

Usage

```r
## S3 method for class 'env_summarized'
print(x, return_area = FALSE, digits = 2, ...)
```

Arguments

- `x` Random patterns.
- `return_area` If true, not the ratio but the area is returned.
- `digits` Number of decimal places (round).
- `...` Arguments passed to `cat`

Details

Printing method for summarized envelope created with `summarize_envelope`.

Value

No return value

See Also

- `summarize_envelope`

Examples

```r
set.seed(42)
input_pattern <- spatstat.random::rThomas(kappa = 15, scale = 0.05, mu = 5)

cluster_env <- spatstat.explore::envelope(input_pattern, fun = "pcf", nsim = 39,
funargs = list(divisor = "d", correction = "Ripley", stoyan = 0.25))

x <- summarize_envelope(cluster_env)
print(x)
```
Description

Local random labelling of marked point pattern

Usage

rlabel_local(X, distance, nsim = 19, drop = TRUE, verbose = TRUE)

Arguments

- **X**: ppp
- **distance**: Mark of points that do not change.
- **nsim**: Number of patterns to simulate.
- **drop**: If nsim = 1 and drop = TRUE, the result will be a point pattern, rather than a list containing a point pattern.
- **verbose**: If TRUE warning messages are printed.

Details

Local random labelling function, i.e. marks will be shuffled only across points within the specified local distance. Technically, this is achieved by sampling the mark of a neighboring point j within the distance d for the focal point i. Thus, the distance d must be selected in a way that each point has at least one neighbor within d.

Returns a list with ppp objects.

Value

list

References


See Also

rlabel
simulate_antecedent_conditions

Examples

```r
set.seed(42)
pattern <- spatstat.random::runifpoint(n = 250, win = spatstat.geom::owin(c(0, 100), c(0, 100)))
spatstat.geom::marks(pattern) <- runif(n = 250, min = 10, max = 120)

rlabel_local(X = pattern, distance = 25, nsim = 19)
```

Description

Simulate heterogenous pattern

Usage

```r
simulate_antecedent_conditions(x, i, j, nsim, heterogenous = FALSE, ...)
```

Arguments

- `x` ppp
- `i` Mark of points that are randomized.
- `j` Mark of points that do not change.
- `nsim` Number of patterns to simulate.
- `heterogenous` If TRUE, points with the mark `i` are randomized using a heterogeneous Poisson process.
- `...` Arguments passed to `spatstat.explore::density.ppp()`.

Details

Simulate point patterns as null model data for `spatstat.explore::envelope()` using antecedent conditions as null model. `x` must be marked point pattern. Antecedent conditions are suitable as a null model if points of type `j` may influence points of type `i`, but not the other way around (Velazquez et al 2016). One example are the positions of seedlings that may be influenced by the position of mature trees.

Returns a list with ppp objects.

Value

- `list`
References


See Also

envelope

Examples

```r
set.seed(42)
pattern_a <- spatstat.random::runifpoint(n = 20)
spatstat.geom::marks(pattern_a) <- "a"
pattern_b <- spatstat.random::runifpoint(n = 100)
spatstat.geom::marks(pattern_b) <- "b"
pattern <- spatstat.geom::superimpose(pattern_a, pattern_b)

null_model <- simulate_antecedent_conditions(x = pattern, i = "b", j = "a", nsim = 19)
spatstat.explore::envelope(Y = pattern, fun = spatstat.explore::pcf, nsim = 19, simulate = null_model)
```

Description

Simulate heterogeneous pattern

Usage

```r
simulate_heterogenous_pattern(x, nsim, fix_n = FALSE, ...)
```

Arguments

- `x` : ppp
- `nsim` : Number of patterns to simulate.
- `fix_n` : Logical if true the null model patterns have exactly the same number of points as input.
- `...` : Arguments passed to spatstat.explore::density.ppp().
Details
Simulate heterogeneous point patterns as null model data for `spatstat.explore::envelope()`.
A heterogeneous Poisson process is used, meaning that there are no interaction between points,
however, the simulated coordinates depend on the intensity $\lambda$ of the input pattern.
Returns a list with `ppp` objects.

Value
list

References

See Also
envelope
density.ppp

Examples
set.seed(42)
input_pattern <- spatstat.random::rpoispp(lambda = function(x, y) {100 * exp(-3 * x)}, nsim = 1)
null_model <- simulate_heterogenous_pattern(input_pattern, nsim = 19)
spatstat.explore::envelope(Y = input_pattern, fun = spatstat.explore::pcf, nsim = 19,
simulate = null_model)

summarize_envelope

Description
Summarize simulation envelope

Usage
summarize_envelope(x, plot_result = FALSE)

Arguments
x fv
plot_result A plot is drawn.
Details

The area above and below the null model envelope is divided by the total area under the curve. If `separated = TRUE`, the first returning value is the relative area above, the second value the relative value below the envelope. If `separated = FALSE` the value is the absolute sum of both ratio. If the value is positive, the area above the envelope is larger than the value below the envelope. If the value is negative, the area under the envelope is larger than the value above the envelope.

The returned `env_summarized` object includes information about the area under the curve where the summary function observed pattern is above or below the null model envelopes.

Value

`env_summarized`

See Also

`envelope`

Examples

```r
set.seed(42)
input_pattern <- spatstat.random::rThomas(kappa = 15, scale = 0.05, mu = 5)

cluster_env <- spatstat.explore::envelope(input_pattern, fun = "pcf", nsim = 39,
funargs = list(divisor = "d", correction = "Ripley", stoyan = 0.25))

summarize_envelope(cluster_env)
```
Index

balance_points, 2

center_l_function, 3

density.ppp, 4, 13

envelope, 8, 12–14
estimate_o_ring, 4
estimate_pcf_fast, 5

Kest, 5

Lest, 3

pcf, 4
pcf.fv, 5
plot.env_summarized, 6
plot_quantums, 7
print.env_summarized, 9

rlabel, 10
rlabel_local, 10

simulate_antecedent_conditions, 11
simulate_heterogenous_pattern, 12
summarize envelope, 6, 7, 9, 13