Package: spatialkernel (via r-universe)

September 7, 2024

Version 0.4-25

Date 2020-02-21

Title Non-Parametric Estimation of Spatial Segregation in a Multivariate Point Process

Copyright 1991-3 Barry Rowlingson and Peter Diggle; 1997-1999 Advanced Interfaces Group, University of Manchester; 1993, 1995, 1997, 1998, 2002, 2005 Jonathan Richard Shewchuk; 1970-2003, Wm. Randolph Franklin; 2011-2017 David Sterratt

URL https://github.com/becarioprecario/spatialkernel

BugReports https://github.com/becarioprecario/spatialkernel/issues

Depends R (>= 2.10)

Imports graphics, grDevices, spatstat.geom, stats

Suggests splancs

Description Edge-corrected kernel density estimation and binary kernel regression estimation for multivariate spatial point process data. For details, see Diggle, P.J., Zheng, P. and Durr, P. A. (2005) <doi:10.1111/j.1467-9876.2005.05373.x>.

License CC BY-NC-SA 4.0

Encoding UTF-8

NeedsCompilation yes

LazyData true

RoxygenNote 7.0.2

Repository https://barryrowlingson.r-universe.dev

RemoteUrl https://github.com/barryrowlingson/spatialkernel

RemoteRef HEAD

RemoteSha beb8395a5b00961d978ccf50a655e15eabd8333b

Contents

`spatialkernel-package` *The Spatialkernel Package*

Description

An R package for spatial point process analysis.

Details

This package contains functions for spatial point process analysis using kernel smoothing methods. This package has been written to be compatible with the **splancs** package which is available on *CRAN* (The Comprehensive R Archive Network).

For a complete list of functions with individual help pages, use `library(help = \ "spatialkernel")`.

Maintainer

Pingping Zheng <`pingping.zheng@lancaster.ac.uk`>

Note

For the convience of the user, we present here examples which show how to use some of the functions in the package.

Author(s)

Pingping Zheng and Peter Diggle

References

1. P. Zheng, P.A. Durr and P.J. Diggle (2004) Edge-correction for Spatial Kernel Smoothing — When Is It Necessary? *Proceedings of the GisVet Conference 2004*, University of Guelph, Ontario, Canada, June 2004.
2. Diggle, P.J., Zheng, P. and Durr, P. A. (2005) Nonparametric estimation of spatial segregation in a multivariate point process: bovine tuberculosis in Cornwall, UK. *J. R. Stat. Soc. C*, **54**, 3, 645–658.

See Also

`cvloglk`, `phat`, `mcseg.test`, `plotphat`, `plotmc`, `pinpoly`, `risk.colors`, `metre`

Examples

```
## An example of spatial segregation analysis
## Not run:
  ## source in Lansing Woods tree data within a polygon boundary
  data(lansing)
  data(polyb)
  ## select data points within polygon
  ndx <- which(pinpoly(polyb, as.matrix(lansing[c("x", "y")])) > 0)
  pts <- as.matrix(lansing[c("x", "y")])[ndx,]
  marks <- lansing[["marks"]][ndx]
  ## select bandwidth
  h <- seq(0.02, 0.1, length=101)
  cv <- cvloglk(pts, marks, h=h)$cv
  hcv <- h[which.max(cv)]
  plot(h, cv, type="l")
  ## estimate type-specific probabilities and do segregation tests
  ## by one integrated function
  sp <- spseg(pts, marks, hcv, opt=3, ntest=99, poly=polyb)
  ## plot estimated type-specific probability surfaces
  plotphat(sp)
  ## additional with pointwise significance contour lines
  plotmc(sp, quan=c(0.025, 0.975))
  ## p-value of the Monte Carlo segregation test
  cat("\np-value of the Monte Carlo segregation test", sp$pvalue)

  ##estimate intensity function at grid point for presentation
  ##with bandwidth hcv
  gridxy <- as.matrix(expand.grid(x=seq(0, 1, length=101), y=seq(0, 1, length=101)))
  ndx <- which(pinpoly(polyb, gridxy) > 0) ##inside point index
  lam <- matrix(NA, ncol=101, nrow=101)
  lam[ndx] <- lambdahat(pts, hcv, gpts = gridxy[ndx,], poly =
      polyb)$lambda
  brks <- pretty(range(lam, na.rm=TRUE), n=12)
  plot(0, 0, xlim=0:1, ylim=0:1, xlab="x", ylab="y", type="n")
  image(x=seq(0, 1, length=101), y=seq(0, 1, length=101),
    z=lam, add=TRUE, breaks=brks, col=risk.colors(length(brks)-1))
  polygon(polyb)
  metre(0, 0.01, 0.05, 0.51, lab=brks, col=risk.colors(length(brks)-1), cex=1)
  ## An example of inhomogeneous intensity function and K function
  ## estimated with the same data
  s <- seq(0, 0.06, length=101)
  lam <- lambdahat(pts, hcv, poly=polyb)$lambda
  kin <- kinhat(pts, lam, polyb, s)
  plot(kin$s, kin$k-pi*(kin$s)^2, xlab="s", ylab="k-pi*s^2", type="l")

## End(Not run)
```

`areapoly` *Signed Area of Polygon*

Description

Calculate the area of a polygon and its boundary direction.

Usage

```
areapoly(poly)
```

Arguments

`poly`	matrix containing the `x,y`-coordinates of the vertices of the polygon boundary.

Value

area positive numeric, the area of the polygon.

sign integer, 1 if the polygon orientation is anticlockwise, -1 otherwise.

poly copy of the passed argument `poly`.

Note

This function is provided here so that users do not need to load other packages, as it is not available in the base R packages.

References

Joseph O'Rourke, Computational Geometry in C (2nd Edition), Cambridge University Press, 2000 edition.

See Also

`metre`, `risk.colors`

`cvloglk` *Cross-Validated Log-Likelihood Function Calculate the cross-validated log-likelihood function.*

Description

Select a common bandwidth for kernel regression estimation of type-specific probabilities of a multivariate Poisson point process with independent component processes of each categorical type by maximizing the cross-validate log-likelihood function.

Usage

```
cvloglk(pts, marks, t = NULL, h)
```

Arguments

`pts`	matrix containing the `x,y`-coordinates of the point locations.
`marks`	numeric/character vector of the marked labels of the type of each point.
`t`	numeric vector of the associated time-periods, default `NULL` for pure spatial data.
`h`	numeric vector of the kernel smoothing bandwidths at which to calculate the cross-validated log-likelihood function.

Details

Select a common bandwidth for kernel regression of type-specific probabilities for all time-periods when the argument `t` is not `NULL`, in which case the data is of a multivariate spatial-temporal point process, with `t` the values of associated time-periods.

Value

A list with components

cv vector of the values of the cross-validated Log-likelihood function.

hcv numeric value which maximizing the cross-validate log-likelihood function

... copy of the arguments `pts, marks, h`.

References

1. Diggle, P.J., Zheng, P. and Durr, P. A. (2005) Nonparametric estimation of spatial segregation in a multivariate point process: bovine tuberculosis in Cornwall, UK. *J. R. Stat. Soc. C*, **54**, 3, 645–658.

See Also

`phat`, `mcseg.test`, and `mcpat.test`

`data` *Lansing Woods Trees*

Description

This is the Lansing Woods Tree data set and an arbitrary polygon boundary within unit square.

Format

`lansing` is an R `data.frame` with `x, y, marks` and `polyb` is an R matrix with `x, y`.

Note

The Lansing Woods tree data set and the arbitrary polygon are provided here for the demonstration of the basic usage of this package. See Section **Examples** in `spatialkernel-package`.

Source

Lansing Woods trees from Lansing Woods, Clinton County, Michigan USA, including hickories, maples, and oaks.

References

Gerrard, D.J. (1969) Competition quotient: a new measure of the competition affecting individual forest trees. *Research Bulletin 20*, Agricultural Experiment Station, Michigan State University.

`filled.contour.poly` *Level (Contour) Plots with Polygonal Boundary This is a revised version of the* **base** R *function* `filled.contour`*. It additionally plots a polygonal boundary.*

Description

Level (Contour) Plots with Polygonal Boundary This is a revised version of the **base** R function `filled.contour`. It additionally plots a polygonal boundary.

Usage

```
filled.contour.poly(
  x = seq(min(poly[, 1]), max(poly[, 1]), len = nrow(z)),
  y = seq(min(poly[, 2]), max(poly[, 2]), len = ncol(z)),
  z,
  poly,
  xlim = range(x, finite = TRUE),
  ylim = range(y, finite = TRUE),
  zlim = range(z, finite = TRUE),
  levels = pretty(zlim, nlevels),
  nlevels = 10,
  color.palette = risk.colors,
  col = color.palette(length(levels) - 1),
  llevels = levels,
  labels = NULL,
  labcex = 0.6,
  drawlabel = TRUE,
  method = "flattest",
  vfont = c("sans serif", "plain"),
  lcol = par("fg"),
  lty = par("lty"),
  lwd = par("lwd"),
  plot.title,
  plot.axes,
  key.title,
  key.axes,
  asp = NA,
  xaxs = "i",
  yaxs = "i",
  las = 1,
  axes = TRUE,
  ...
)
```

Arguments

`x`	locations of grid lines at which the values in `z` are measured. These must be in ascending order. By default, equally spaced values from 0 to 1 are used. If `x` is a list, its components `x$x` and `x$y` are used for `x` and `y`, respectively. If the list has component `z` this is used for `z`.
`y`	See `x` arg.
`z`	a matrix containing the values to be plotted (`NA`s are allowed). Note that `x` can be used instead of `z` for convenience.
`poly`	a matrix containing the `x,y`-coordinates of the vertices of the polygon boundary.
`xlim`	`x` limits for the plot.
`ylim`	`y` limits for the plot.
`zlim`	`z` limits for the plot.
`levels`	a set of levels which are used to partition the range of `z`. Must be strictly increasing (and finite). Areas with `z` values between consecutive levels are painted with the same color.
`nlevels`	if `levels` is not specified, the range of `z` is divided into approximately this many levels.
`color.palette`	a color palette function used to assign colors in the plot.
`col`	an explicit set of colors to be used in the plot. This argument overrides any palette function specification.

`llevels`	numeric vector of levels at which to draw contour lines, default is the same as `levels`.
`labels`	a vector giving the labels for the contour lines. If `NULL` then the levels are used as labels.
`labcex`	`cex` for contour labelling.
`drawlabel`	logical, contour lines are labelled if `TRUE`.
`method`	character string specifying where the labels will be located. Possible values are "simple", "edge" and "flattest" (the default). See the Details section.
`vfont`	if a character vector of length 2 is specified, then Hershey vector fonts are used for the contour labels. The first element of the vector selects a typeface and the second element selects a fontindex (see `text` for more information).
`lcol`	color for the lines drawn.
`lty`	line type for the lines drawn.
`lwd`	line width for the lines drawn.
`plot.title`	statement which add title to the main plot.
`plot.axes`	statement which draws axes on the main plot. This overrides the default axes.
`key.title`	statement which adds title to the plot key.
`key.axes`	statement which draws axes on the plot key. This overrides the default axis.
`asp`	the `y/x` aspect ratio, see `plot.window`.
`xaxs`	the `x` axis style. The default is to use internal labeling.
`yaxs`	the `y` axis style. The default is to use internal labeling.
`las`	the style of labeling to be used. The default is to use horizontal labeling.
`axes`	Logical. Should axes be drawn? See `plot.default`.
`...`	additional graphical parameters.

Note

By defining `z` values as `NA` at points outside the polygonal boundary, `filled.contour.poly` produces a contour plot within the polygonal boundary.

See Also

`filled.contour`, `contour` and `pinpoly`

kinhat — *Inhomogeneous K-function Estimation Estimate the inhomogeneous K function of a non-stationary point pattern.*

Description

Inhomogeneous K-function Estimation Estimate the inhomogeneous K function of a non-stationary point pattern.

Usage

```
kinhat(pts, lambda, poly, s)
```

Arguments

`pts`	matrix of the `x,y`-coordinates of the point locations.
`lambda`	intensity function evaluated at the above point locations.
`poly`	matrix of the `x,y`-coordinates of the polygon boundary.
`s`	vector of distances at which to calculate the K function.

Details

The inhomogeneous K function is a generalization of the usual K function defined for a second-order intensity-reweighted stationary point process, proposed by Baddeley *et\ al* (2000).

When the true intensity function is unknown, and is to be estimated from the same data as been used to estimate the K function, a modified kernel density estimation implemented in `lambdahat` with argument `gpts=NULL` can be used to calculate the estimated intensity at data points. See Baddeley *et al* (2000) for details, and Diggle, P.J., *et al* (2006) for a cautious note.

Value

A list with components

k values of estimated K at the distances `s`.

s copy of `s`.

Note

This code is adapted from **splancs** (Rowlingson and Diggle, 1993) fortran code for the estimation of homogeneous K function `khat`, with edge correction inherited for a general polygonal area.

References

1. Baddeley, A. J. and M?ller, J. and Waagepetersen R. (2000) Non and semi-parametric estimation of interaction in inhomogeneous point patterns, *Statistica Neerlandica*, **54**, 3, 329–350.
2. Diggle, P.J., V. Gómez-Rubio, P.E. Brown, A.G. Chetwynd and S. Gooding (2006) Second-order analysis of inhomogeneous spatial point processes using case-control data, *submitted to Biometrics*.
3. Rowlingson, B. and Diggle, P. (1993) Splancs: spatial point pattern analysis code in S-Plus. *Computers and Geosciences*, **19**, 627–655.

See Also

khat, lambdahat

---

lambdahat *Kernel Density Estimation of Intensity Function*

Description

Kernel density estimation of the intensity function of a two-dimensional point process.

Usage

```
lambdahat(pts, h, gpts = NULL, poly = NULL, edge = TRUE)
```

Arguments

pts	matrix containing the x,y-coordinates of the data point locations.
h	numeric value of the bandwidth used in the kernel smoothing.
gpts	matrix containing the x,y-coordinates of point locations at which to calculate the intensity function, usually a fine grid points within poly, default NULL to estimate intensity function at data locations.
poly	matrix containing the x,y-coordinates of the vertices of the polygon boundary in an anticlockwise order.
edge	logical, with default TRUE to do edge-correction.

Details

Kernel smoothing methods are widely used to estimate the intensity of a spatial point process. One problem which arises is the need to handle edge effects. Several methods of edge-correction have been proposed. The adjustment factor proposed in Berman and Diggle (1989) is a double integration $\int_A K[(x-x_0)/h]/h^2$, where A is a polygonal area, K is the smoothing kernel and h is the bandwidth used for the smoothing. Zheng, P. *et\ al* (2004) proposed an algorithm for fast calculate of Berman and Diggle's adjustment factor.

When gpts is NULL, lambdahat uses a leave-one-out estimator for the intensity at each of the data points, as been suggested in Baddeley *et al* (2000). This leave-one-out estimate at each of the data points then can be used in the inhomogeneous K function estimation `kinhat` when the true intensity function is unknown.

The default kernel is the *Gaussian*. The kernel function is selected by calling `setkernel`.

Value

A list with components

lambda numeric vector of the estimated intensity function.

... copy of the arguments `pts, gpts, h, poly, edge`.

Note

In principle, the *double adaptive* double integration algorithm of Zheng, P. *et\ al* (2004) can be applied to other kernel functions. Furthermore, the area at the present is enclosed by a simple polygon which could be generalized into a complex area with polygonal holes inside. For instance, a large lake lays within the land area of study.

Other source codes used in the implementation of the double integration algorithm include

- Laurie, D.P. (1982) *adaptive cubature* code in Fortran;
- Shewchuk, J.R. *triangulation* code in C;
- Alan Murta's *polygon intersection* code in C (*Project: Generic Polygon Clipper*).

References

1. M. Berman and P. Diggle (1989) Estimating weighted integrals of the second-order intensity of a spatial point process, *J. R. Stat. Soc. B*, **51**, 81–92.
2. P. Zheng, P.A. Durr and P.J. Diggle (2004) Edge–correction for Spatial Kernel Smoothing — When Is It Necessary? *Proceedings of the GisVet Conference 2004*, University of Guelph, Ontario, Canada, June 2004.
3. Baddeley, A. J. and M&#248;ller, J. and Waagepetersen R. (2000) Non and semi-parametric estimation of interaction in inhomogeneous point patterns, *Statistica Neerlandica*, **54**, 3, 329–350.
4. Laurie, D.P. (1982). Algorithm 584 CUBTRI: Adaptive Cubature over a Triangle. *ACM–Trans. Math. Software*, **8**, 210–218.
5. Jonathan R. Shewchuk, *Triangle, a Two-Dimensional Quality Mesh Generator and Delaunay Triangulator* at `http://www-2.cs.cmu.edu/~quake/triangle.html`.
6. Alan Murta, *General Polygon Clipper* at `http://www.cs.man.ac.uk/~toby/gpc`.
7. NAG's Numerical Library. *Chapter 11: Quadrature*, NAG's Fortran 90 Library. `https://www.nag.co.uk/numeric/fn/manual/html/c11_fn04.html`

See Also

`setkernel`, `kinhat`, `density`

`mcpat.test`	*Monte Carlo Inference of Temporal Changes in Spatial Segregation An approximate Monte Carlo test of temporal changes in a multivariate spatial-temporal point process.*

Description

Monte Carlo Inference of Temporal Changes in Spatial Segregation An approximate Monte Carlo test of temporal changes in a multivariate spatial-temporal point process.

Usage

```
mcpat.test(pts, marks, t, h, ntest = 100, proc = TRUE)
```

Arguments

`pts`	matrix containing the `x,y`-coordinates of the data point locations.
`marks`	numeric/character vector of the marked type labels of the data points.
`t`	numeric vector of the associated time-periods.
`h`	numeric vector of the bandwidths at which to calculate the cross-validated log-likelihood function pooled over times.
`ntest`	integer with default 100, number of simulations for the Monte Carlo test
`proc`	logical, default `TRUE` prints the processing messages.

Details

The spatial-temporal data are denoted as (x_i, m_i, t_i), where x_i are the spatial locations, m_i are the categorical mark sequence numbers, and t_i are the associated time-periods.

The null hypothesis is that the type-specific probability surfaces are constant over time-periods, *i.e.*, $p_k(x,t) = p_k(x)$, for any t, where $p_k(x,t)$ are the type-specific probabilities for kth category within time-period t.

Each Monte Carlo simulation is sampled from an approximate *true* type-specific probability surfaces — the estimated one from the data. Approximately, the simulated data and the original data are samples from the same probability distribution under the null hypothesis. See Diggle, P.J. *et al* (2005) for more details.

Value

A list with components

pvalue p-value of the approximate Monte Carlo test.

... copy of `pts, marks, t, h, ntest`.

References

1. Diggle, P. J. and Zheng, P. and Durr, P. A. (2005) Nonparametric estimation of spatial segregation in a multivariate point process: bovine tuberculosis in Cornwall, UK. *J. R. Stat. Soc. C*, **54**, 3, 645–658.

See Also

`cvloglk`, `phat`, `mcseg.test`

`mcseg.test` *Monte Carlo Test of Spatial Segregation in Multivariate Point Process*

Description

Monte Carlo test of spatial segregation in a multivariate point process by simulating data from random re-labelling of the categorical marks.

Usage

```
mcseg.test(pts, marks, h, stpts = NULL, ntest = 100, proc = TRUE)
```

Arguments

`pts`	matrix containing the `x,y`-coordinates of the data point locations.
`marks`	numeric/character vector of the marked type labels of the point pattern.
`h`	numeric vector of the bandwidths at which to calculate the cross-validated likelihood function.
`stpts`	matrix containing the `x,y`-coordinates of the locations at which to implement the pointwise segregation test, with default `NULL` not to do the pointwise segregation test.
`ntest`	integer with default 100, number of simulations for the Monte Carlo test.
`proc`	logical with default `TRUE` to print the processing messages.

Details

The null hypothesis is that the estimated risk surface is spatially constant, *i.e.*, the type-specific probabilities are $p_k(x) = p_k$, for all k, see `phat`. Each Monte Carlo simulation is done by relabeling the data categorical marks at random whilst preserving the observed number of cases of each type.

The segregation test can also be done pointwise, usually at a fine grid of points, to mark the areas where the estimated type-specific probabilities are significantly greater or smaller than the spatial average.

Value

A list with components

pvalue numeric, p-value of the Monte Carlo test.

stpvalue matrix, p-values of the test at each point in `stpts` (if `stpts` is not `NULL`), with each column corresponds to one type

... copy of the arguments `pts`, `marks`, `h`, `stpts`, `ntest`, `proc`.

References

1. Kelsall, J. E. and Diggle, P. J. (1998) Spatial variation in risk: a nonparametric binary regression approach, *Applied Statistics*, **47**, 559–573.
2. Diggle, P. J. and Zheng, P. and Durr, P. A. (2005) Nonparametric estimation of spatial segregation in a multivariate point process: bovine tuberculosis in Cornwall, UK. *J. R. Stat. Soc. C*, **54**, 3, 645–658.

See Also

`cvloglk` and `phat`

`metre` *Plot Color Level Metre*

Description

This is a simple function provided here for the convenience of users. It adds a key showing how the colors map to the values of an `image` plot.

Usage

```
metre(
  xl,
  yb,
  xr,
  yt,
  lab,
  cols = risk.colors(length(lab) - 1),
  shift = 0,
  cex = 1
)
```

Arguments

`xl`	Left coordinate of the color level metre bar.
`yb`	Bottom coordinate of the color level metre bar.
`xr`	Right coordinate of the color level metre bar.
`yt`	Top coordinate of the color level metre bar.
`lab`	metre level labels in the metre.
`cols`	associated colours, defaults to use `risk.colors`.
`shift`	distance to shift the label texts away from the metre bar.
`cex`	numeric character expansion factor.

See Also

`risk.colors`

`package.version` *Listing Loaded/Installed Package Versions List version numbers of loaded or installed packages.*

Description

Listing Loaded/Installed Package Versions List version numbers of loaded or installed packages.

Usage

```
package.version(all.available = FALSE, lib.loc = NULL)
```

Arguments

`all.available`	logical, if `TRUE` return all available packages.
`lib.loc`	character vector describing the location of R library trees to search through, or `NULL`. The default value of `NULL` corresponds to all libraries currently known.

Details

This is a revised version of the **base** R function `.package`. It gives both package names and their version numbers.

Value

A list with components

package names of loaded or available packages if `all.available` is `TRUE`.

version associated package versions.

See Also

`.packages`

phat *Estimate Type-Specific Probabilities*

Description

Estimate the type-specific probabilities for a multivariate Poisson point process with independent component processes of each type.

Usage

```
phat(gpts, pts, marks, h)
```

Arguments

`gpts`	matrix containing the `x,y`-coordinates of the point locations at which type-specific probabilities are estimated.
`pts`	matrix containing the `x,y`-coordinates of the data points.
`marks`	numeric/character vector of the types of the point in the data.
`h`	numeric value of the bandwidth used in the kernel regression.

Details

The type-specific probabilities for data (x_i, m_i), where x_i are the spatial point locations and m_i are the categorical mark sequence numbers, $m_i = 1, 2, \ldots$, are estimated using the kernel smoothing methodology $\hat{p}_k(x) = \sum_{i=1}^n w_{ik}(x) I(m_i = k)$, where $w_{ik}(x) = w_k(x - x_i) / \sum_{j=1}^n w_k(x - x_j)$, $w_k(.)$ is the kernel function with bandwidth $h_k > 0$, $w_k(x) = w_0(x/h_k)/h_k^2$, and $w_0(\cdot)$ is the standardised form of the kernel function.

The default kernel is the *Gaussian*. Different kernels can be selected by calling `setkernel`.

Value

A list with components

p matrix of the type-specific probabilities for all types, with the type marks as the matrix row names.

... copy of the arguments `pts`, `dpts`, `marks`, `h`.

References

1. Diggle, P. J. and Zheng, P. and Durr, P. A. (2005) Nonparametric estimation of spatial segregation in a multivariate point process: bovine tuberculosis in Cornwall, UK. *J. R. Stat. Soc. C*, **54**, 3, 645–658.

See Also

`cvloglk`, `mcseg.test`, and `setkernel`

`pinpoly` *Check if Points are within Polygon*

Description

Check the location of point(s) with respect to a polygon.

Usage

```
pinpoly(poly, pts)
```

Arguments

`poly`	matrix containing the `x,y`-coordinates of the vertices of the polygon boundary.
`pts`	matrix of containing the `x,y`-coordinates of the point locations.

Value

An integer vector of indicators for each point in `pts`,

-1 error when number of polygon vertices exceeds 3000;

0 outside the polygon;

1 at the polygon boundary;

2 inside the polygon.

Note

This function is provided here so that users do not need to load other packages, as it is not available in the **base** R packages. THE VERTICES MAY BE LISTED CLOCKWISE OR ANTICLOCK-WISE

The return values have been changed from the original ones so that the point is inside (including at the boundary) if positive.

References

This Fortran code comes from Wm Randolph Franklin, Electrical, Computer, and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, New York, at website `https://wrf.ecse.rpi.edu/nikola/pages/`.

See Also

`phat` and `mcseg.test`

risk.colors *Color Palette*

Description

This color palette is designed to show risk levels with different colours from small risk (bright orange) to high risk (dark red).

Usage

```
risk.colors(n)
```

Arguments

n	number of colors (>= 1) to be in the palette.

See Also

`metre`, `colors`, and `palette`.

Examples

```
## risk pie with ten levels
pie(rep(1,10), labels = seq(0.1, 1, 0.1), col = risk.colors(10))
```

setkernel *Select Smoothing Kernel Function*

Description

Select a kernel function for kernel regression and kernel smoothing.

Usage

```
setkernel(kernel = NULL)
```

Arguments

kernel	character string giving the smoothing kernel to be used. This must be one of *gaussian*, *epanechnikov*, *quadratic*, *quartic*, or `NULL`, and may be abbreviated to a unique prefix.

Value

A character string of the kernel function selected, or the kernel function currently being used when `kernel` is `NULL`.

Note

The default kernel used is *Gaussian*. Unless users want to use a non-default kernel, there is no need to call `setkernel`. *quadratic* is an alias for *epanechnikov*.

`setkernel` setup kernel function for both kernel regression in the type-specific probability estimation and the kernel smoothing in the intensity function estimation.

See Also

`cvloglk`, `phat` and `lambdahat`

Examples

```
## Not run:
  setkernel("e") ## Select "epanechnikov" kernel
  setkernel()    ## show the kernel currrently being used

## End(Not run)
```

`spseg.matrix` *Integrated Functions for Spatial Segregation Analysis Spatial segregation analysis to be performed by a single function and presentations by associated* `plot` *functions.*

Description

Run the spatial segregation analysis

spseg for spatstat objects

Usage

```
## S3 method for class 'matrix'
spseg(
  pts,
  marks,
  h,
  opt = 2,
  ntest = 100,
  poly = NULL,
  delta = min(apply(apply(pts, 2, range), 2, diff))/100,
  proc = TRUE,
  ...
)

plotcv(obj, ...)

plotphat(
  obj,
  types = unique(obj$marks),
  sup = TRUE,
  col = risk.colors(10),
  breaks = seq(0, 1, length = length(col) + 1),
  ...
)

plotmc(
  obj,
  types = unique(obj$marks),
  quan = c(0.05, 0.95),
  sup = FALSE,
  col = risk.colors(10),
  breaks = seq(0, 1, length = length(col) + 1),
  ...
)

spseg(pts, ...)

## S3 method for class 'ppp'
spseg(pts, h, opt, ...)
```

Arguments

`pts`	an object that contains the points. This could be a two-column matrix or a ppp object from spatstat.
`marks`	numeric/character vector of the types of the point in the data.
`h`	numeric vector of the kernel smoothing bandwidth at which to calculate the cross-validated log-likelihood function.
`opt`	integer, 1 to select bandwidth; 2 to calculate type-specific probabilities; and 3 to do the Monte Carlo segregation test.
`ntest`	integer with default 100, number of simulations for the Monte Carlo test.
`poly`	matrix containing the `x,y`-coordinates of the polygonal boundary of the data.
`delta`	spacing distance of grid points at which to calculate the estimated type-specific probabilities for `image` plot.
`proc`	logical with default `TRUE` to print the processing message.
`...`	other arguments concerning `plot` and `points`
`obj`	list of the returning value of `spseg`.
`types`	numeric/character types of the marks of data points to plot the estimated type-specific probabilities, default to plot all types.
`sup`	logical with default `FALSE`, if `TRUE` to superimpose data points on the estimated type-specific probability surface.
`col`	list of colors such as that generated by `risk.colors`.
`breaks`	a set of breakpoints for the `col`: must give one more breakpoint than colour.
`quan`	numeric, the pointwise significance levels to add contours to `image` plot of the estimated type-specific probability surface, with default of `c(0.05, 0.95)`.

Details

`spseg` implements a complete spatial segregation analysis by selecting bandwidth, calculating the type-specific probabilities, and then carrying out the Monte Carlo test of spatial segregation and pointwise significance. Some `plot` functions are also provided here so that users can easily present the results.

These functions are provided only for the convenience of users. Users can instead use individual functions to implement the analysis step by step and plot the diagrams as they wish.

Examples of how to use `spseg` and present results using `plot` functions are presented in `spatialkernel-package`.

This is the details of the S3 generic method

Does spseg for marked ppp objects

Value

A list with components

hcv bandwidth selected by the cross-validated log-likelihood function.

gridx,gridy `x, y` coordinate vectors at which the grid points are generated at which to calculate the type-specific probabilities and pointwise segregation test *p*-value.

p estimated type-specific probabilities at grid points generated by vectors `gridx, gridy`.

pvalue *p*-value of the Monte Carlo spatial segregation test.

stpvalue pointwise *p*-value of the Monte Carlo spatial segregation test.

... copy of `pts, marks, h, opt`.

spseg results

an spseg object

Note

Setting `h` to a unique value may force `spseg` to skip the selecting bandwidth step, go straight to calculate the type-specific probabilities and then test the spatial segregation with this fixed value of bandwidth.

Author(s)

Barry Rowlingson

Barry Rowlingson

See Also

`cvloglk`, `phat`, `mcseg.test`, `pinpoly`, `risk.colors`, and `metre`

Index