Package 'kernstadapt'

May 8, 2026

Title Adaptive Kernel Estimators for Point Process Intensities on Linear Networks

Version 0.4.0

Maintainer Jonatan A González <jonathan.gonzalez@kaust.edu.sa>

Description Adaptive estimation of the first-order intensity function of a spatio-temporal point process using kernels and variable bandwidths. The methodology used for estimation is presented in González and Moraga (2022). <doi:10.48550/arXiv.2208.12026>.

License MIT + file LICENSE

Language en-GB

Encoding UTF-8

RoxygenNote 7.3.2

Imports misc3d, sparr, spatstat.explore, spatstat.univar, spatstat.geom, spatstat.random, spatstat.utils, spatstat.linnet

Suggests knitr, rmarkdown, ggplot2

VignetteBuilder knitr

Depends R (>= 4.0)

NeedsCompilation no

Author Jonatan A González [aut, cre] (ORCID: <https://orcid.org/0000-0002-2296-5271>), Paula Moraga [aut] (ORCID: <https://orcid.org/0000-0001-5266-0201>)

Repository CRAN

Date/Publication 2024-09-26 11:20:05 UTC

Contents

`aegiss` *Diggle et al.'s data: Non-specific gastrointestinal data*

Description

A spatio-temporal point pattern of locations and times of individuals with non-specific gastrointestinal infections in Hampshire, UK, from 2001 to 2003.

Usage

```
data("aegiss")
```

Format

An object of class `"ppp"` Entries include

`x`	Cartesian x-coordinate of infection
`y`	Cartesian y-coordinate of infection
`marks`	An integer vector of values indicating
the time of infection starting in 1 `window`	Cartesian coordinates of Hampshire's map

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

Source

Diggle, P. (2015) AEGISS1. Syndromic surveillance of gastro-intestinal illness https://www.research.lancs.ac.uk/portal/en/datasets/aegiss1-syndromic-surveillance-of-gastrointestinal-illness(.html

References

Diggle, L. Knorr-Held, R. B, T. Su, P. Hawtin, and T. Bryant. (2003) On-line monitoring of public health surveillance data. In B. R. and S. D.F., editors, Monitoring the Health of Populations: *Statistical Principles and Methods for Public Health Surveillance.* **233-266**. Oxford University Press.

Examples

```
data(aegiss)
head(aegiss)
plot(aegiss, bg = rainbow(250), pch = 21)
```

---

amazon *Locations of fires in the Amazon biome.*

Description

A spatio-temporal point pattern of locations and times of active deforestation fires (starting within past 24 hours) from 01/01/2021 to 10/10/2021 (284 days).

Usage

```
data("amazon")
```

Format

An object of class "ppp" Entries include

x	Cartesian x-coordinate of fire
y	Cartesian y-coordinate of fire
marks	An integer vector of values indicating
the time of the fire starting in 2 window	Coordinates of Amazonia biome map

See ppp.object for details of the format.

Source

The Amazon Dashboard (https://www.globalfiredata.org/)

Examples

```
data(amazon)
head(amazon)
# Plot a sample
X <- amazon[sample.int(amazon$n, 3000)]
plot(X, bg = rainbow(250), pch = 21, main = "Amazon fires")
```

bw.abram.net *Abramson's adaptive bandwidth for networks*

Description

Computes adaptive smoothing bandwidth in the network case according to the inverse-square-root rule of Abramson (1982).

Usage

```
## S3 method for class 'net'
bw.abram(
  X,
  h0,
  ...,
  at = c("points", "pixels"),
  hp = h0,
  pilot = NULL,
  trim = 5,
  smoother = densityQuick.lpp
)
```

Arguments

X	A point pattern on a linear network (object of class "lpp").
h0	The global smoothing bandwidth. The default is the maximal oversmoothing principle of Terrell (1990).
...	Additional arguments passed to smoother to control the type of smoothing.
at	Character string specifying whether to compute bandwidths at the points (at = "points", the default) or to compute bandwidths at every bin in a bin grid (at = "bins").
hp	Optional. A scalar pilot bandwidth, used for estimation of the pilot density if required.
pilot	Optional. A pilot estimation of the intensity to plug in Abramson's formula.
trim	A trimming value to cut extreme large bandwidths.
smoother	Smoother for the pilot. A function or character string, specifying the function to be used to compute the pilot estimate when pilot is NULL or is a point pattern.

Details

This function returns a set of adaptive smoothing bandwidths driven by Abramson's (1982) method for a point pattern in a linear network. The bandwidth at location u is given by

$$\epsilon(u) = h0 * \min\left[\frac{1}{\gamma}\sqrt{\frac{n}{\tilde{lambda}(u)}}, \texttt{trim}\right]$$

where $\tilde{\lambda}(u)$ is a pilot estimate of the network varying intensity and γ is a scaling constant depending on the pilot estimate.

Value

If `at = "points"` (the default), the result is a numeric vector with one bandwidth for each data point in `X`. If `at = "pixels"`, the output is an object with class "linim"

Author(s)

Jonatan A. González

References

Abramson, I. (1982) On bandwidth variation in kernel estimates — a square root law. *Annals of Statistics*, **10**(4), 1217-1223.

Examples

```
#To be done
```

`bw.abram.temp` *Abramson's adaptive temporal bandwidths*

Description

Computes adaptive smoothing bandwidth in the temporal case according to the inverse-square-root rule of Abramson (1982).

Usage

```
## S3 method for class 'temp'
bw.abram(X, h0 = NULL, ..., nt = 128, trim = NULL, at = "points")
```

Arguments

`X`	A vector (a temporal point pattern) from which the bandwidths should be computed.
`h0`	The global smoothing bandwidth. The default is Silverman's rule of thumb (bw.nrd0).
`...`	Additional arguments passed to smoother to control the type of smoothing.
`nt`	The number of equally spaced points at which the temporal density is to be estimated.
`trim`	A trimming value to cut extreme large bandwidths.
`at`	Character string specifying whether to compute bandwidths at the points (at = "points", the default) or to compute bandwidths at every bin in a bin grid (at = "bins").

Details

This function returns a set of temporal adaptive smoothing bandwidths driven by the methods of Abramson (1982) and Hall and Marron (1988). The bandwidth at location v is given by

$$\delta(v) = h0 * \min\left[\frac{1}{\gamma}\sqrt{\frac{n}{\lambda^{\mathrm{t}}(v)}}, \texttt{trim}\right]$$

where $\lambda^{\mathrm{t}}(v)$ is a pilot estimate of the temporally varying intensity and γ is a scaling constant depending on the pilot estimate.

Value

If `at = "points"` (the default), the result is a numeric vector with one bandwidth for each data point in `X`. If `at = "bins"`, the output is an object with class "density" where `y` component is a vector with the estimated intensity values (see density).

Author(s)

Jonatan A. González

References

Abramson, I. (1982) On bandwidth variation in kernel estimates — a square root law. *Annals of Statistics*, **10**(4), 1217-1223.

Davies, T.M. and Baddeley, A. (2018) Fast computation of spatially adaptive kernel estimates. *Statistics and Computing*, **28**(4), 937-956.

Davies, T.M., Marshall, J.C., and Hazelton, M.L. (2018) Tutorial on kernel estimation of continuous spatial and spatiotemporal relative risk. *Statistics in Medicine*, **37**(7), 1191-1221.

Hall, P. and Marron, J.S. (1988) Variable window width kernel density estimates of probability densities. *Probability Theory and Related Fields*, **80**, 37-49.

Silverman, B.W. (1986) *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, New York.

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes https://arxiv.org/pdf/2208.12026

Examples

```
t <- 2 * rbeta(100, 1.5, 5.5, 0.2) #Simulated temporal point pattern
bw.abram.temp(t)
```

`dens.direct` *Direct estimation of non-separable adaptive spatio-temporal intensity estimator*

Description

Provides an adaptive-bandwidth kernel estimate for spatio-temporal point patterns in a non-separable fashion by calculating the classical estimator, i.e., the slowest estimation.

Usage

```
dens.direct(
  X,
  t = NULL,
  dimyx = 128,
  dimt = 128,
  bw.xy = NULL,
  bw.t = NULL,
  at = c("bins", "points")
)
```

Arguments

`X`	A spatial point pattern (an object of class `ppp`) with the spatial coordinates of the observations. It may contain marks representing times.
`t`	A numeric vector of temporal coordinates with equal length to the number of points in `X`. This gives the time associated with each spatial point. This argument is not necessary if time marks are provided to the point pattern `X`.
`dimyx`	Spatial pixel resolution. The default is 128 for each axes.
`dimt`	Temporal bin vector dimension. The default is 128.
`bw.xy`	Numeric vector of spatial smoothing bandwidths for each point in `X`. By default this is computed using bw.abram, with `h0` given by OS.
`bw.t`	Numeric vector of temporal smoothing bandwidths for each point in `t`. By default this is computed using bw.abram.temp.
`at`	String specifying whether to estimate the intensity at a mesh (`at = "bins"`) or only at the points of `X` (`at = "points"`).

Details

This function computes a non-separable spatio-temporal adaptive kernel estimate of the intensity. It starts from a planar point pattern `X` and a vector of times `t` and apply a non-separable kernel estimator for each of the points of `X`. The arguments `bw.xy` and `bw.t` specify the smoothing bandwidth vectors to be applied to each of the points in `X` and `t`. They should be a numeric vectors of bandwidths.

Value

If `at = "points"` (the default), the result is a numeric vector with one entry for each data point in `X`. if `at = "bins"` is a list named (by time-point) list of pixel images (im objects) corresponding to the joint spatio-temporal intensity over space at each discretised time bin.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

Examples

```
data(lGCpp)
X <- lGCpp[sample.int(200)] # A random subset
stIntensity <- dens.direct(X, dimyx = 16, dimt = 4)
plot(spatstat.geom::as.solist(stIntensity), ncols = 4,
     main = 'Non-separable direct example', equal.ribbon = TRUE)


data(aegiss)
X <- aegiss[sample.int(500)] # A random subset
stIntensity <- dens.direct(X,
                           dimyx = 32, dimt = 16,
                           at = "bins")
plot(spatstat.geom::as.imlist(stIntensity[12:15]),
     main = 'Non-separable direct example')
```

`dens.direct.sep` *Direct separable adaptive spatio-temporal intensity estimator*

Description

Provides an adaptive-bandwidth kernel estimate for spatio-temporal point patterns in a separable fashion, i.e., by multiplying spatial and temporal marginals. This estimation is performed by calculating the classical estimator, i.e., the slowest estimation.

Usage

```
dens.direct.sep(
  X,
  t = NULL,
  dimyx = 128,
```

```
  dimt = 128,
  bw.xy = NULL,
  bw.t = NULL,
  at = c("bins", "points")
)
```

Arguments

X	A spatial point pattern (an object of class ppp) with the spatial coordinates of the observations. It may contain marks representing times.
t	A numeric vector of temporal coordinates with equal length to the number of points in X. This gives the time associated with each spatial point. This argument is not necessary if time marks are provided to the point pattern X.
dimyx	Spatial pixel resolution. The default is 128 for each axes.
dimt	Temporal bin vector dimension. The default is 128.
bw.xy	Numeric vector of spatial smoothing bandwidths for each point in X. By default this is computed using bw.abram.
bw.t	Numeric vector of temporal smoothing bandwidths for each point in t. By default this is computed using bw.abram.temp.
at	String specifying whether to estimate the intensity at a mesh (at = "bins") or only at the points of X (at = "points").

Details

This function computes a spatio-temporal adaptive kernel estimate of the intensity in a separable fashion. It starts from a planar point pattern X and a vector of times t and uses a direct estimator for each dimension, then it multiplies both components and normalises by the number of points to preserve the mass. The arguments bw.xy and bw.t specify the smoothing bandwidth vectors to be applied to each of the points in X and t. They should be a numeric vectors of bandwidths. The method partition the range of bandwidths into intervals, subdividing the points of the pattern X and t into sub-patterns according to the bandwidths, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying ngroups.xy = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy) in the spatial case and ngroups.t = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy).

Value

If at = "points", the result is a numeric vector with one entry for each data point in X. if at = "bins" is a list named (by time-point) list of pixel images (im objects) corresponding to the joint spatio-temporal intensity over space at each discretised time bin.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

Examples

```
data(santander)
X <- santander[sample.int(200)]
stIntensity <- dens.direct.sep(X,
                               dimyx = 32, dimt = 6,
                               at = "bins")
plot(spatstat.geom::as.solist(stIntensity[2:4]),
     main = 'Direct separable example', equal.ribbon = TRUE)
```

`dens.net.EqualSplit` *Adaptive linear netwokr intensity estimator using the Okabe-Sugihara equal-split algorithms*

Description

Computes an adaptive-bandwidth kernel estimate for the intensity function through the Okabe-Sugihara equal-split algorithms by using binning of the bandwidth values.

Usage

```
dens.net.EqualSplit(
  X,
  ...,
  weights = NULL,
  bw = NULL,
  ngroups = NULL,
  at = c("pixels", "points"),
  verbose = FALSE
)
```

Arguments

`X`	A point pattern on a linear network (an object of class `lpp`) to be smoothed.
`...`	Extra arguments passed to densityHeat.lpp.
`weights`	Optional. Numeric vector of weights associated with the points of X. Weights can be positive, negative or zero.
`bw`	Numeric vector of spatial smoothing bandwidths for each point in `X`. By default this is computed using bw.abram.
`ngroups`	Number of groups in which the bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the spatial part with a bandwidth selected as the median of the bw.xy vector.
`at`	String specifying whether to estimate the intensity at a mesh (`at = "pixels"`) or only at the points of `X` (`at = "points"`).
`verbose`	Logical value indicating whether to print progress reports for every partition group.

Details

This function computes an adaptive kernel estimate of the intensity on linear networks. It starts from a point pattern `X` and partition the spatial component to apply a kernel estimator within each cell. The argument `bw` specifies the smoothing bandwidth vector to be applied to each of the points in `X`. It should be a numeric vector of bandwidths. The method partition the range of bandwidths into intervals, subdividing the points of the pattern `X` into sub-patterns according to the bandwidths, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying `ngroups = 1` is the same as fixed-bandwidth smoothing with bandwidth `sigma = median(bw)`.

Value

If `at = "points"` (the default), the result is a numeric vector with one entry for each data point in `X`. if `at = "pixels"` is a pixel image on a linear network (linim objects) corresponding to the intensity over linear network.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

`dens.net.heat` *Adaptive linear network intensity estimator based on heat kernel*

Description

Provides an adaptive-bandwidth kernel estimate for point patterns on linear networks by using binning of the bandwidth values.

Usage

```
dens.net.heat(
  X,
  ...,
  weights = NULL,
  bw = NULL,
  ngroups = NULL,
  at = c("pixels", "points")
)
```

Arguments

X	A point pattern on a linear network (an object of class lpp) to be smoothed.
...	Extra arguments passed to densityHeat.lpp.
weights	Optional. Numeric vector of weights associated with the points of X. Weights can be positive, negative or zero.
bw	Numeric vector of spatial smoothing bandwidths for each point in X. By default this is computed using bw.abram.
ngroups	Number of groups in which the bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the spatial part with a bandwidth selected as the median of the bw.xy vector.
at	String specifying whether to estimate the intensity at a mesh (at = "pixels") or only at the points of X (at = "points").

Details

This function computes an adaptive kernel estimate of the intensity on linear networks. It starts from a point pattern X and partition the component to apply a kernel estimator within each cell. The argument bw specifies the smoothing bandwidth vector to be applied to each of the points in X. It should be a numeric vector of bandwidths. The method partition the range of bandwidths into intervals, subdividing the points of the pattern X into sub-patterns according to the bandwidths, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying ngroups = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw).

Value

If at = "points" (the default), the result is a numeric vector with one entry for each data point in X. if at = "pixels" is a pixel image on a linear network (linim objects) corresponding to the intensity over linear network.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

---

dens.par *Non-separable adaptive spatio-temporal intensity estimator*

Description

Provides an adaptive-bandwidth kernel estimate for spatio-temporal point patterns in a non-separable fashion by using binning of the bandwidth values.

Usage

```
dens.par(
  X,
  t = NULL,
  dimyx = 128,
  dimt = 128,
  bw.xy = NULL,
  bw.t = NULL,
  ngroups.xy = NULL,
  ngroups.t = NULL,
  at = c("bins", "points")
)
```

Arguments

X	A spatial point pattern (an object of class ppp) with the spatial coordinates of the observations. It may contain marks representing times.
t	A numeric vector of temporal coordinates with equal length to the number of points in X. This gives the time associated with each spatial point. This argument is not necessary if time marks are provided to the point pattern X.
dimyx	Spatial pixel resolution. The default is 128 for each axes.
dimt	Temporal bin vector dimension. The default is 128.
bw.xy	Numeric vector of spatial smoothing bandwidths for each point in X. By default this is computed using bw.abram.
bw.t	Numeric vector of temporal smoothing bandwidths for each point in t. By default this is computed using bw.abram.temp.
ngroups.xy	Number of groups in which the spatial bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the spatial part with a bandwidth selected as the median of the bw.xy vector.
ngroups.t	Number of groups in which the temporal bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the temporal part with a bandwidth selected as the median of the bw.t vector.
at	String specifying whether to estimate the intensity at a mesh (at = "bins") or only at the points of X (at = "points").

Details

This function computes a non-separable spatio-temporal adaptive kernel estimate of the intensity. It starts from a planar point pattern X and a vector of times t and partition (cells) the spatial and temporal components to apply a non-separable kernel estimator within each cell. The arguments bw.xy and bw.t specify the smoothing bandwidth vectors to be applied to each of the points in X and t. They should be a numeric vectors of bandwidths. The method partition the range of bandwidths into intervals, subdividing the points of the pattern X and t into sub-patterns according to the bandwidths, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying ngroups.xy = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy) in the spatial case and ngroups.t = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy).

Value

If `at = "points"` (the default), the result is a numeric vector with one entry for each data point in `X`. if `at = "bins"` is a list named (by time-point) list of pixel images (im objects) corresponding to the joint spatio-temporal intensity over space at each discretised time bin.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

Examples

```
data(lGCpp)
stIntensity <- dens.par(lGCpp, dimt = 16)
plot(spatstat.geom::as.solist(stIntensity[13:16]), ncols = 4,
     main = 'Non-separable Example', equal.ribbon = TRUE)
```

`dens.par.sep` *Separable adaptive spatio-temporal intensity estimator*

Description

Provides an adaptive-bandwidth kernel estimate for spatio-temporal point patterns in a separable fashion, i.e., by multiplying spatial and temporal marginals.

Usage

```
dens.par.sep(
  X,
  t = NULL,
  dimyx = 128,
  dimt = 128,
  bw.xy = NULL,
  bw.t = NULL,
  ngroups.xy = NULL,
  ngroups.t = NULL,
  at = c("bins", "points")
)
```

Arguments

X	A spatial point pattern (an object of class ppp) with the spatial coordinates of the observations. It may contain marks representing times.
t	A numeric vector of temporal coordinates with equal length to the number of points in X. This gives the time associated with each spatial point. This argument is not necessary if time marks are provided to the point pattern X.
dimyx	Spatial pixel resolution. The default is 128 for each axes.
dimt	Temporal bin vector dimension. The default is 128.
bw.xy	Numeric vector of spatial smoothing bandwidths for each point in X. By default this is computed using bw.abram.
bw.t	Numeric vector of temporal smoothing bandwidths for each point in t. By default this is computed using bw.abram.temp.
ngroups.xy	Number of groups in which the spatial bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the spatial part with a bandwidth selected as the median of the bw.xy vector.
ngroups.t	Number of groups in which the temporal bandwidths should be partitioned. If this number is 1, then a classical non-adaptive estimator will be used for the temporal part with a bandwidth selected as the median of the bw.t vector.
at	String specifying whether to estimate the intensity at a mesh (at = "bins") or only at the points of X (at = "points").

Details

This function computes a spatio-temporal adaptive kernel estimate of the intensity in a separable fashion. It starts from a planar point pattern X and a vector of times t and uses partition techniques separately for the spatial and temporal components, then it multiplies both components and normalises by the number of points to preserve the mass. The arguments bw.xy and bw.t specify the smoothing bandwidth vectors to be applied to each of the points in X and t. They should be a numeric vectors of bandwidths. The method partition the range of bandwidths into intervals, subdividing the points of the pattern X and t into sub-patterns according to the bandwidths, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying ngroups.xy = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy) in the spatial case and ngroups.t = 1 is the same as fixed-bandwidth smoothing with bandwidth sigma = median(bw.xy).

Value

If at = "points" (the default), the result is a numeric vector with one entry for each data point in X. if at = "bins" is a list named (by time-point) list of pixel images (im objects) corresponding to the joint spatio-temporal intensity over space at each discretised time bin.

Author(s)

Jonatan A. González

References

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

Examples

```
data(santander)
stIntensity <- dens.par.sep(santander,
                            dimt = 16,
                            ngroups.xy = 3, ngroups.t = 2,
                            at = "bins")
plot(spatstat.geom::as.solist(stIntensity[12:15]), ncols = 4,
     main = 'Separable Example', equal.ribbon = TRUE)
```

`dens.par.temp` *Adaptive kernel estimate of intensity of a temporal point pattern*

Description

Estimates the intensity of a point process with only temporal dimension by applying an adaptive (variable bandwidth) Gaussian edge-corrected kernel smoothing.

Usage

```
dens.par.temp(
  t,
  dimt = 128,
  bw.t = NULL,
  ngroups.t = NULL,
  at = c("bins", "points")
)
```

Arguments

`t`	Temporal point pattern, a vector with observations.
`dimt`	Bin vector dimension. The default is 128.
`bw.t`	Numeric vector of smoothing bandwidths for each point in t. The default is to compute bandwidths using bw.abram.temp.
`ngroups.t`	Number of groups into which the bandwidths should be partitioned and discretised. The default is the square root (rounded) of the number of points of `t`.
`at`	String specifying whether to estimate the intensity at bins points (`at = "bins"`) or only at the points of t (`at = "points"`).

Details

This function computes a temporally-adaptive kernel estimate of the intensity from a one-dimensional point pattern t using the partitioning technique of Davies and Baddeley (2018). The argument bw.t specifies the smoothing bandwidths to be applied to each of the points in X. It should be a numeric

vector of bandwidths. Let the points of t be $t_1, ..., t_n$ and the corresponding bandwidths $\sigma_1, ..., \sigma_n$, then the adaptive kernel estimate of intensity at a location v is

$$\lambda(v) = \sum_{i=1}^{n} \frac{K(v, t_i; \sigma_i)}{c(t; \sigma_i)}$$

where $K()$ is the Gaussian smoothing kernel. The method partition the range of bandwidths into ngroups.t intervals, correspondingly subdividing the points of the pattern `t` into ngroups.t sub-patterns according to bandwidth, and applying fixed-bandwidth smoothing to each sub-pattern. Specifying `ngroups.t = 1` is the same as fixed-bandwidth smoothing with bandwidth `sigma = median(bw.t)`.

Value

If `at = "points"` (the default), the result is a numeric vector with one entry for each data point in `t`. If `at = "bins"` the result is a data.frame containing the x, y coordinates of the intensity function.

Author(s)

Jonatan A. González

References

Davies, T.M. and Baddeley, A. (2018) Fast computation of spatially adaptive kernel estimates. Statistics and Computing, 28(4), 937-956.

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes http://arxiv.org/pdf/2208.12026

Examples

```
t <- rbeta(100, 1,4,0.8)
tIntensity <- dens.par.temp(t, at = "bins")
plot(tIntensity$x, tIntensity$y, type = "l")
```

`lGCpp` *Simulated spatio-temporal log Gaussian Cox point pattern data*

Description

A simulated spatio-temporal log Gaussian Cox point pattern with a underlying non-separable Geneiting covariance.

Usage

```
data("lGCpp")
```

Format

An object of class "ppp" Entries include

x	Cartesian x-coordinates
y	Cartesian y-coordinates
marks	A vector of values indicating the times
window	Cartesian coordinates of the unit square

See ppp.object for details of the format.

References

Gabriel E, Rowlingson B, Diggle PJ (2013). stpp: An R Package for Plotting, Simulating and Analyzing Spatio-Temporal Point Patterns. *Journal of Statistical Software,* **(53)(2)**(2), 1–29.

González J.A. and Moraga P. (2018) An adaptive kernel estimator for the intensity function of spatio-temporal point processes. <arXiv:2208.12026>

Examples

```
data(lGCpp)
head(lGCpp)
plot(lGCpp, bg = rainbow(250), pch = 21, cex = 1)
```

---

santander *Santander earthquakes epicentres*

Description

A spatio-temporal point pattern of locations and times of earthquake epicentres in Santander-Colombia with a magnitude above 3.5ML, recorded from January 2010 to December 2020.

Usage

```
data("santander")
```

Format

An object of class "ppp" Entries include

x	Cartesian x-coordinate of infection
y	Cartesian y-coordinate of infection
marks	An integer vector of values indicating the time of the earthquake starting in 1
window	Cartesian coordinates of Santander's map

See ppp.object for details of the format.

Source

Colombian national geologic institute, https://www.sgc.gov.co/

References

González et. al., (2021) Classification of Events Using Local Pair Correlation Functions for Spatial Point Patterns. *Journal of Agricultural, Biological and Environmental Statistics.* **26(4)** 538-559.

Examples

```
data(santander)
head(santander)
plot(santander, bg = rainbow(250), pch = 21, cex = 1)
```

---

separability.test *Separability test for spatio-temporal point processes*

Description

Performs a separability test of the first-order intensity function based on a Fisher Monte Carlo test of cell counts.

Usage

```
separability.test(X, t = NULL, nx = NULL, ny = NULL, nt = NULL, nperm = 1000)
```

Arguments

X	A spatial point pattern (an object of class ppp) with the spatial coordinates of the observations.
t	A numeric vector of temporal coordinates with equal length to the number of points in X. This gives the time associated with each spatial point.
nx, ny, nt	Numbers of quadrats in the x, y and t directions.
nperm	An integer specifying the number of replicates used in the Monte Carlo test.

Details

This function performs a basic test of the separability hypothesis in a manner similar to independence test in two-way contingency tables. The test is conditional on the observed number of points. It considers a regular division of the interval T into disjoint sub-intervals $T_1, ..., T_{n_t}$ and similarly a division of the window W into disjoint subsets $W_1, ..., W n_x \times n_y$. Then the function computes Fisher's test statistic and get a p-value based on Monte Carlos approximation.

Value

A list with class "htest" containing the following components:

`p.value`	the approximate p-value of the test.
`method`	the character string "Separability test based on Fisher's for counting data".
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Note

This is a fast preliminary separability test.

Author(s)

Jonatan A. González

References

Ghorbani et al. (2021) Testing the first-order separability hypothesis for spatio-temporal point patterns, *Computational Statistics & Data Analysis,* **161**, p.107245.

Examples

```
data(lGCpp)
separability.test(lGCpp, nx = 5, ny = 4, nt = 3, nperm = 500)


data(aegiss)
separability.test(aegiss, nx = 8, ny = 8, nt = 4)
```

Index