Package 'btb'

March 17, 2025

Encoding UTF-8

Type Package

Title Beyond the Border - Kernel Density Estimation for Urban Geography

Description The kernelSmoothing() function allows you to square and smooth geolocated data. It calculates a classical kernel smoothing (conservative) or a geographically weighted median. There are four major call modes of the function.
The first call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth) for a classical kernel smoothing and automatic grid.
The second call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, quantiles) for a geographically weighted median and automatic grid.
The third call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, centroids) for a classical kernel smoothing and user grid.
The fourth call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, quantiles, centroids) for a geographically weighted median and user grid.
Geographically weighted summary statistics : a framework for localised exploratory data analysis, C.Brunsdon & al., in Computers, Environment and Urban Systems C.Brunsdon & al. (2002) <doi:10.1016/S0198-9715(01)00009-6>,
Statistical Analysis of Spatial and Spatio-Temporal Point Patterns, Third Edition, Diggle, pp. 83-86, (2003) <doi:10.1080/13658816.2014.937718>.

Version 0.2.1

License GPL (>= 2)

Date 2025-03-13

Imports methods, Rcpp (>= 1.0.9), sf, RcppParallel, magrittr

Suggests knitr, rmarkdown, testthat (>= 3.0.0),

Depends R (>= 3.3.0), dplyr, mapsf

Maintainer Solène Colin <solene.colin@insee.fr>

LinkingTo Rcpp, RcppParallel, BH (>= 1.60.0-1), RcppArmadillo

LazyData true

URL https://github.com/InseeFr/btb, https://inseefr.github.io/btb/

BugReports https://github.com/InseeFr/btb/issues

RoxygenNote 7.3.2

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation yes

Author Arlindo Dos Santos [aut],
François Sémécurbe [aut],
Julien Pramil [aut],
Solène Colin [cre, ctb],
Kim Antunez [ctb],
Auriane Renaud [ctb],
Farida Marouchi [ctb],
Joachim Timotéo [ctb],
Institut national de la statistique et des études économiques [cph]

Repository CRAN

Date/Publication 2025-03-17 11:30:07 UTC

Contents

btb 2
btb_add_centroids 3
btb_add_inspire 4
btb_ptsToGrid 5
btb_smooth 6
constituerGrappes 10
constituerMatriceEffectifs 11
dfPrix_SP95_2016 12
dfRestaurantParis 12
dfToGrid 13
kernelSmoothing 14
pixel_france 15
reunion 15

Index **17**

`btb` *Beyond the Border - Kernel Density Estimation for Urban Geography*

Description

The kernelSmoothing() function allows you to square and smooth geolocated data. It calculates a classical kernel smoothing (conservative) or a geographically weighted median. There are four major call modes of the function. The first call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth) for a classical kernel smoothing and automatic grid. The second call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, quantiles) for a geographically weighted median and automatic

grid. The third call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, centroids) for a classical kernel smoothing and user grid. The fourth call mode is kernelSmoothing(obs, epsg, cellsize, bandwidth, quantiles, centroids) for a geographically weighted median and user grid.

`btb_add_centroids` *Link points to their centroids*

Description

- Link some points to their centroids in a grid segmentation
- Relie des points aux centroides des carreaux auxquels ces points appartiennent (dans un découpage de l'espace en grille carroyée)

Usage

```
btb_add_centroids(
  pts,
  iCellSize,
  offset = c(0L, 0L),
  names_coords = c("x", "y"),
  names_centro = c("x_centro", "y_centro"),
  add = TRUE
)
```

Arguments

`pts` : points (`df` of `sf` object)

`iCellSize` :

- Size of the square cells (**meters**)
- Taille des côtés des carreaux (**mètres**)

`offset` (**numeric vector of size 2**)

- Offset for a grid non centered on the geographical referential origin
- Décalage si utilisation d'une grille non centrée sur l'origine du référentiel géographique

`names_coords` :

- Names of the latitude-longitude variables (**character vector**)
- Noms des variables de latitude-longitude (**vecteur character**)

`names_centro`

- Names of the latitude-longitude variables for added centroids (**character vector**)
  - Noms des variables de latitude-longitude pour les centroïdes ajoutés (**vecteur character**)

`add` (**boolean**)

- If TRUE : returns pts + centroids coordinates
- Si TRUE : retourne pts + les coordonnées des centroïdes

Details

Works with sf points but only with coordinates in meters (and not degrees !). Do not use sf points with GPS coordinates for example.

Value

- `pts` table with additional centroids coordinates `x_centro` and `y_centro` (`df` of `sf` object)
- Table `pts` avec les coordonnées des centroïdes `x_centro` and `y_centro` (objet `df` of `sf`)

Examples

```
pts <- data.frame(
x = c(656913.1 , 348296.3 , 842276.3 , 716750.0 , 667418.2),
y = c(6855995 , 6788073 , 6385680 , 7003984 , 6585793),
val = 1:5)
btb_add_centroids(pts, 100, names_centro = c("centroX", "centroY"))
btb_add_centroids(pts, 100, offset = c(50, 50), names_centro = c("centroX", "centroY"))
pts2 <- sf::st_as_sf(pts, coords = c("x","y"), crs = 2154)
btb_add_centroids(pts2, 50)
```

`btb_add_inspire` *Creates Inpire names for a grid*

Description

Creates Inpire names for a grid defined with :

- centroids coordinates
- Squares size(s)
- Projection system

Ajoute les noms des identifiants en norme Inspire des carreaux définis à partir :

- des coordonnées de leurs centroides
- de la taille de leur côté en mètres
- d'un système de projection)

Usage

```
btb_add_inspire(
  pts_centro,
  sEPSG,
  iCellSize,
  names_centro = c("x_centro", "y_centro")
)
```

Arguments

pts_centro	(**df object**) : table of centroids.
sEPSG	(**integer or character**) : epsg code.
iCellSize	(**integer**) : cells size(s) in meters. Can be a vector for irregular grids
names_centro	(**character vector**) : vector of names for longitude/latitude variables. Default c("x_centro","y_centro").

Value

(**df**) pts_centro table with additional *idInspire* variable

Examples

```
pts_centro <- data.frame(x_centro = c(100, 100, 300, 300, 500),
y_centro = c(100, 300, 100, 300, 100))
btb_add_inspire(pts_centro, sEPSG = 2154, iCellSize = 200)
```

---

btb_ptsToGrid *Compute a grid from centroid points*

Description

Function to compute a grid (regular or not) from centroid points.

(Fonction permettant de générer une grille (régulière ou non) à partir de centroïdes)

Usage

```
btb_ptsToGrid(
  pts,
  sEPSG = NA,
  iCellSize = NULL,
  names_centro = c("x_centro", "y_centro"),
  inspire = FALSE
)
```

Arguments

pts	A simple `data.frame` with the centroids coordinates of the squares to draw, or a `sf` object of centroides. To generate an irregular grid, a column with each cell size must be provided and named `iCellSize`. (Un simple `data.frame` comportant les coordonnées des carrés à dessiner, ou un objet `sf` des centroides. Pour obtenir une grille irrégulière, il faut fournir une colonne indiquant la taille de chaque carreau, et nommée `iCellSize`.

sEPSG	EPSG code of projection (character). For example, the RGF93 / Lambert-93 projection has "2154" code. (code EPSG de la projection (character). Par exemple, la projection RGF93 / Lambert-93 a pour code "2154".)
iCellSize	Cell size of the grid. If this argument is provided, the grid is regular. (Taille des carreaux de la grille. Si cet argument est fourni, la grille est régulière.)
names_centro	(**character vector**) • vector of names for longitude/latitude variables. Default c("x_centro","y_centro"). • vecteur des noms des variables de longitude/latitude. Par défaut : c("x_centro","y_centro")
inspire	(boolean) : if TRUE, returns a column for Inspire grid names.

Value

Returns an object of class sf and data.frame.

(Retourne un objet de classe sf et data.frame.)

Examples

```
# example 1 - regular grid
pts <- data.frame(x_centro = c(100, 100, 300, 300, 500),
y_centro = c(100, 300, 100, 300, 100))
carResult <- btb_ptsToGrid(pts = pts, sEPSG = "2154", iCellSize = 200)
# write_sf(obj = carResult, dsn = "regularGrid.shp", delete_layer = TRUE)

# example 2 - irregular grid
pts <- data.frame(x = c(50, 50, 150, 150, 300)
                  , y = c(50, 150, 50, 150, 100)
                  , iCellSize = c(50, 50, 50, 50, 100))
carResult <- btb_ptsToGrid(pts = pts, sEPSG = "2154",names_centro=c("x","y"))
# write_sf(obj = carResult, dsn = "irregularGrid.shp", delete_layer = TRUE)
# Exemple 3 : sf points (no epsg)
pts <- data.frame(x = c(100, 100, 300, 300, 500), y = c(100, 300, 100, 300, 100))
pts <- sf::st_as_sf(pts,coords=c("x","y"))
carResult <- btb_ptsToGrid(pts = pts, sEPSG = "2154", iCellSize = 200)
# Exemple 3 : sf points (no epsg)
pts <- data.frame(x = c(100, 100, 300, 300, 500),
y = c(100, 300, 100, 300, 100))
pts <- sf::st_as_sf(pts,coords=c("x","y"),crs=2154)
carResult <- btb_ptsToGrid(pts = pts, sEPSG = "2154", iCellSize = 200)
```

---

btb_smooth *Smoothing with a bisquare kernel or median*

Description

Smoothing function with a bisquare kernel or median.

(Fonction de lissage à partir d'un noyau bisquare ou de la médiane.)

Usage

```
btb_smooth(
  pts,
  sEPSG = NA,
  iCellSize = NA,
  iBandwidth,
  vQuantiles = NULL,
  dfCentroids = NULL,
  iNeighbor = NULL,
  inspire = FALSE,
  iNbObsMin = 250
)
```

Arguments

pts	A data.frame with cartesian geographical coordinates and variables to smooth. (x, y, var1, var2, ...) (Un data.frame comportant les coordonnées géographiques cartésiennes (x,y), ainsi que les variables que l'on souhaite lisser. (x, y, var1, var2, ...)
sEPSG	EPSG code of projection (character). For example, the RGF93 / Lambert-93 projection has "2154" code. (code EPSG de la projection (character). Par exemple, la projection RGF93 / Lambert-93 a pour code "2154".)‘
iCellSize	Cell size of the grid (integer). The unit of measurement is free. It must be the same as the unit of iBandwidth variable. (Taille des carreaux (integer). Le choix de l'unité de mesure est laissé libre à l'utilisateur. Elle doit seulement être la même que celle de la variable iBandwidth.)
iBandwidth	Radius of the Kernel Density Estimator (integer). This bandwidth acts as a smoothing parameter, controlling the balance between bias and variance. A large bandwidth leads to a very smooth (i.e. high-bias) density distribution. A small bandwidth leads to an unsmooth (i.e. high-variance) density distribution. The unit of measurement is free. It must be the same as the unit of iCellSize variable. (Rayon de lissage de l'estimation d'intensité par noyau (integer). Cette bande-passante se comporte comme un paramètre de lissage, controlant l'équilibre entre biais et variance. Un rayon élevé conduit à une densité tres lissée, avec un biais élevé. Un petit rayon génère une densité peu lissée avec une forte variance. Le choix de l'unité de mesure est laissé libre à l'utilisateur. Elle doit seulement être la même que celle de la variable iCellSize.
vQuantiles	Percentile vector to calculate. For example c(0.1, 0.25, 0.5) will calculate the first decile, the first quartile and the median. (Vecteur des quantiles à calculer. Par exemple c(0.1, 0.25, 0.5) retournera le premier décile, le premier quartile et la mediane.)‘
dfCentroids	A data.frame with two columns (x, y) containing coordinates of the user's centroids. The coordinates must be in the same projection than (pts).

	(Un `data.frame` avec deux colonnes (x, y) contenant les coordonnées des centroides de l'utilisateur. Les coordonnées doivent être dans le même système de coordonnées que (`pts`).)
`iNeighbor`	Technical parameter, leave empty. (`integer`) (Paramètre technique pour calculer l'étendue des points d'estimations, à ne pas remplir. (`integer`))
`inspire`	(boolean) : if TRUE, returns a column for Inspire grid names.
`iNbObsMin`	Minimum size of constituted grappes for median smoothing. (`integer`) (Taille minimale des grappes constituées pour le lissage "médian" (géographiquement pondéré). (`integer`))

Details

Returns an object inheriting from the `data.frame` class. (Retourne un objet qui se comporte comme un `data.frame`, par heritage.)

- Smoothing covers a set of methods to extract pertinent and structuring information from noisy data. In the field of spatial analysis, and most widely in quantitative geography, smoothing is used to modelise density variations of a population distribution in geographical space. Kernel smoothing methods are widely used. In this method, for each location x, we count the number of events of a process within a distance h of x, and weighted by the square reciprocal of the radius h. We apply a edge-correction to deal with edge-effects. So the method is conservative..
- Le lissage recouvre un ensemble de méthodes pour extraire d'une source de données bruitées une information pertinente et structurante. Dans le champ de l'analyse spatiale et plus largement de la géographie quantitative, le lissage est principalement utilisé pour modéliser les variations de densités d'une distribution de population dans l'espace géographique. On utilise principalement des méthodes de lissage par noyau. Il s'agit ici, pour chaque point x, de comptabliser le nombre d' "évènements" d'un processus à une distance h de ce point, tout en ponderant ce nombre par l'inverse de la distance h au carré. On applique une correction à la ponderation afin de traiter les effets de bord. Cette méthode est conservative.

Value

Returns an object inheriting from the `data.frame` class. (Retourne un objet qui se comporte comme un `data.frame`, par heritage.)

References

- "Geographically weighted summary statistics : a framework for localised exploratory data analysis", C.Brunsdon & al., in Computers, Environment and Urban Systems 2002
- Statistical Analysis of Spatial and Spatio-Temporal Point Patterns, Third Edition, Diggle, 2003, pp. 83-86

Examples

```
# Example 1
data(dfPrix_SP95_2016)
dfPrix_SP95_2016$nbObs <- 1L
```

```
dfSmoothed <- btb::btb_smooth(pts = dfPrix_SP95_2016,
                              sEPSG = "2154",
                              iCellSize = 5000L,
                              iBandwidth = 30000L,
                              inspire = TRUE)
dfSmoothed$prix95 <- dfSmoothed$SP95 / dfSmoothed$nbObs * 100
library(mapsf)
mf_map(dfSmoothed,
      type = "choro",
      var = "prix95",
      breaks = "fisher",
      nbreaks = 5,
      border = NA,
      leg_title = "prix du SP95 en centimes")
# Example 2
data(reunion)
# Call mode 1: classic smoothing - automatic grid
reunionSmoothed <- btb_smooth( pts = reunion,
                                   sEPSG = "32740",
                                   iCellSize = 200L,
                                   iBandwidth = 400L)
library(mapsf)
mf_map(reunionSmoothed,
      type = "choro",
      var = "houhold",
      breaks = "fisher",
      nbreaks = 5,
      border = NA)
# Call mode 2: median smoothing - automatic grid
reunionSmoothed <- btb_smooth( pts = reunion,
                                    sEPSG = "32740",
                                    iCellSize = 200L,
                                    iBandwidth = 400L,
                                    vQuantiles = c(0.1, 0.5, 0.9))
mf_map(reunionSmoothed,
      type = "choro",
      var = "houhold_05",
      breaks = "fisher",
      nbreaks = 5,
      border = NA)
# Call mode 3: classic smoothing - user grid
dfCentroidsUser <- merge( x = seq(from =  314400L, to =  378800L, by = 200L),
                          y = seq(from = 7634000L, to = 7691200L, by = 200L))
reunionSmoothed <- btb_smooth( pts = reunion,
                                   sEPSG = "32740",
                                   iCellSize = 200L,
                                   iBandwidth = 400L,
                                   dfCentroids = dfCentroidsUser)
reunionSmoothed <- reunionSmoothed[reunionSmoothed$houhold > 0, ]
mf_map(reunionSmoothed,
      type = "choro",
      var = "houhold",
      breaks = "fisher",
```

```
      nbreaks = 5,
      border = NA)
# Call mode 4: median smoothing - user grid
reunionSmoothed <- btb_smooth( pts = reunion,
                                    sEPSG = "32740",
                                    iCellSize = 200L,
                                    iBandwidth = 400L,
                                    vQuantiles = c(0.1, 0.5, 0.9),
                                    dfCentroids = dfCentroidsUser)
reunionSmoothed <- reunionSmoothed[reunionSmoothed$nbObs > 0, ]
mf_map(reunionSmoothed,
      type = "choro",
      var = "houhold_05",
      breaks = "fisher",
      nbreaks = 5,
      border = NA)
```

`constituerGrappes` *constituerGrappes*

Description

Quadtree variant function.

(Variante de la fonction Quadtree.)

Usage

```
constituerGrappes(iNbObsMin, mEffectifs)
```

Arguments

`iNbObsMin`	• An `integer` representing the minimum number of elements in each cluster – Un `data.frame` représentant le nombre minimum d'éléments dans chaque grappe.)
`mEffectifs`	• staffing matrix for each cell (`matrix`). – matrice des effectifs pour chaque case (`matrix`). .)

Value

- Returns a `matrix` with the cluster number for each cell .
- Retourne une `matrix` avec le numéro de grappe pour chaque cellule.

Author(s)

- Psar Analyse Urbaine Insee
- Arlindo Dos Santos
- Francois Semecurbe

`constituerMatriceEffectifs` *constituerMatriceEffectifs*

Description

Function constituting a membership matrix (Fonction constituant une matrice des effectifs)

Usage

```
constituerMatriceEffectifs(vLigneObservation, vColonneObservation)
```

Arguments

`vLigneObservation` :

- A `vector` containing the line number of each observation
- (Un `vector` contenant le numéro de ligne de chaque observation.)

`vColonneObservation` :

- A `vector` containing the column number of each observation
- (Un `vector` contenant le numéro de colonne de chaque observation.)

Value

Returns a `matrix` with the number of observations for each cell.

(Retourne une `matrix` avec le nombre d'observations pour chaque cellule.)

Author(s)

- Psar Analyse Urbaine Insee
- Arlindo Dos Santos
- Francois Semecurbe

dfPrix_SP95_2016 *Unleaded 95 price in France in 2016*

Description

Unleaded 95 price in France in 2016 (prix du sans-plomb 95 en France en 2016)

Usage

```
dfPrix_SP95_2016
```

Format

`dfPrix_SP95_2016`**:**
A data frame with 5,573 rows and 3 columns:

x longitude
y latitude
SP95 Unleaded price - prix du sans plomb

Source

https://www.prix-carburants.gouv.fr/rubrique/opendata/

dfRestaurantParis *Parisian restaurants*

Description

Parisian restaurants (Restaurants parisiens)

Usage

```
dfRestaurantParis
```

Format

`dfRestaurantParis`**:**
A data frame with 13,823 rows and 17 columns:

x longitude
y latitude
french French restaurant
asian asian restaurant
caribbean caribbean restaurant
maghreb maghreb restaurant

african african restaurant
european european restaurant
south_america south_american restaurant
indian indian restaurant
world world food restaurant
cafeteria cafeteria restaurant
tea_house tea_house
brasserie brasserie restaurant
fastfood fastfood restaurant
bar bar
restaurant restaurant

Source

source : https://opendata.paris.fr

`dfToGrid` *Depreciated function of btb_ptsToGrid*

Description

Function dfToGrid has been replaced by btb_ptsToGrid

Usage

```
dfToGrid(df, sEPSG, iCellSize = NULL)
```

Arguments

`df`	centroid points
`sEPSG`	epsg code
`iCellSize`	cell size

Value

a grid

`kernelSmoothing` *Depreciated function of btb_smooth*

Description

Function kernelSmoothing has been replaced by `btb_smooth`

Usage

```
kernelSmoothing(
  dfObservations,
  sEPSG,
  iCellSize,
  iBandwidth,
  vQuantiles = NULL,
  dfCentroids = NULL,
  fUpdateProgress = NULL,
  iNeighbor = NULL,
  iNbObsMin = 250
)
```

Arguments

`dfObservations`	points
`sEPSG`	epsg code
`iCellSize`	cell size
`iBandwidth`	smoothing bandwidth
`vQuantiles`	quantiles
`dfCentroids`	user grid
`fUpdateProgress`	message parameter
`iNeighbor`	number of neighbors pixels
`iNbObsMin`	technical parameter

Value

a warning message

pixel_france *France grid with 1km square tiles*

Description

Whole France grid with 1km square tiles. Lambert 93 projection.

(Grille France entière avec des carreaux d'1km de côté. Projection Lambert 93)

Usage

```
pixel_france
```

Format

`pixel_france`**:**

A data frame with 54,8495 rows and 2 columns:

x longitude of pixel's centroids

y latitude of pixel's centroids

Source

source : https://opendata.paris.fr

reunion *Households of Reunion*

Description

Households of Reunion (Menages de La Reunion) Gridded database with a grid cell resolution of 200 meters. (Données carroyées à 200 mètres).

Usage

```
reunion
```

Format

`reunion`**:**

A data frame with 14,076 rows and 4 columns:

x longitude

y latitude

houhold number of households - nombre de ménages

phouhold number of poor households - nombre de ménages pauvres

Source

Source : Insee, Localized data : Fiscal Revenue - 31/12/2010 and Housing Tax - 01/01/2011

Source : Insee, Revenus Fiscaux Localisés (RFL) au 31 decembre 2010 et Taxe d'habitation (TH) au 1er janvier 2011)

Index

∗ **datasets**
dfPrix_SP95_2016, 12
dfRestaurantParis, 12
pixel_france, 15
reunion, 15

btb, 2
btb_add_centroids, 3
btb_add_inspire, 4
btb_ptsToGrid, 5
btb_smooth, 6

constituerGrappes, 10
constituerMatriceEffectifs, 11

dfPrix_SP95_2016, 12
dfRestaurantParis, 12
dfToGrid, 13

kernelSmoothing, 14

pixel_france, 15

reunion, 15