Package 'sfhotspot'

February 19, 2025

Title Hot-Spot Analysis with Simple Features

Version 0.9.1

Description Identify and understand clusters of points (typically representing the locations of places or events) stored in simple-features (SF) objects. This is useful for analysing, for example, hot-spots of crime events. The package emphasises producing results from point SF data in a single step using reasonable default values for all other arguments, to aid rapid data analysis by users who are starting out. Functions available include kernel density estimation (for details, see Yip (2020) <doi:10.22224/gistbok/2020.1.12>), analysis of spatial association (Getis and Ord (1992) <doi:10.1111/j.1538-4632.1992.tb00261.x>) and hot-spot classification (Chainey (2020) ISBN:158948584X).

License MIT + file LICENSE

Language en-GB

URL http://pkgs.lesscrime.info/sfhotspot/

BugReports https://github.com/mpjashby/sfhotspot/issues

Encoding UTF-8

RoxygenNote 7.3.2

Imports cli, ggplot2, rlang, sf, SpatialKDE, spdep, tibble

Depends R (>= 3.5)

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

LazyData true

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Matt Ashby [aut, cre] (<https://orcid.org/0000-0003-4201-9239>)

Maintainer Matt Ashby <matthew.ashby@ucl.ac.uk>

Repository CRAN

Date/Publication 2025-02-19 18:10:02 UTC

Contents

`autoplot.hspt_c` *Plot map of hotspot classifications*

Description

Plot the output produced by `hotspot_classify` with reasonable default values.

Usage

```
## S3 method for class 'hspt_c'
autoplot(object, ...)
```

Arguments

`object`	An object with the class `hspt_c`, e.g. as produced by `hotspot_classify`.
`...`	Currently ignored, but may be used for further options in future.

Value

A `ggplot` object.

This function returns a `ggplot` object, meaning you can further control the appearance of the plot by adding calls to further `ggplot2` functions.

autoplot.hspt_d *Plot map of changes in grid counts*

Description

Plot the output produced by hotspot_change with reasonable default values.

Usage

```
## S3 method for class 'hspt_d'
autoplot(object, ...)

## S3 method for class 'hspt_d'
autolayer(object, ...)
```

Arguments

object	An object with the class hspt_d, e.g. as produced by hotspot_change.
...	Currently ignored, but may be used for further options in future.

Value

A ggplot object.

This function returns a ggplot object, meaning you can further control the appearance of the plot by adding calls to further ggplot2 functions.

Functions

- autolayer(hspt_d): Create a ggplot layer of change in grid counts

autoplot.hspt_k *Plot map of kernel-density values*

Description

Plot the output produced by hotspot_kde with reasonable default values.

Usage

```
## S3 method for class 'hspt_k'
autoplot(object, ...)

## S3 method for class 'hspt_k'
autolayer(object, ...)
```

Arguments

object	An object with the class hspt_k, e.g. as produced by hotspot_kde.
...	further arguments passed to geom_sf, e.g. alpha.

Value

A ggplot object or layer that can be used as part of a ggplot stack.

autoplot returns a ggplot object, meaning you can further control the appearance of the plot by adding calls to further ggplot2 functions.

Functions

- autolayer(hspt_k): Create a ggplot layer of kernel-density values

---

autoplot.hspt_n *Plot map of grid counts*

Description

Plot the output produced by hotspot_count with reasonable default values.

Usage

```
## S3 method for class 'hspt_n'
autoplot(object, ...)

## S3 method for class 'hspt_n'
autolayer(object, ...)
```

Arguments

object	An object with the class hspt_n, e.g. as produced by hotspot_count.
...	further arguments passed to geom_sf, e.g. alpha.

Value

A ggplot object or layer that can be used as part of a ggplot stack.

autoplot returns a ggplot object, meaning you can further control the appearance of the plot by adding calls to further ggplot2 functions.

Functions

- autolayer(hspt_n): Create a ggplot layer of grid counts

hotspot_change *Identify change in hotspots over time*

Description

Identify change in the number of points (typically representing events) between two periods (before and after a specified date) or in two groups (e.g. on weekdays or at weekends).

Usage

```
hotspot_change(
  data,
  time = NULL,
  boundary = NULL,
  groups = NULL,
  cell_size = NULL,
  grid_type = "rect",
  grid = NULL,
  quiet = FALSE
)
```

Arguments

data	sf data frame containing points.
time	Name of the column in data containing Date or POSIXt values representing the date associated with each point. Ignored if groups is not NULL. If this argument is NULL and data contains a single column of Date or POSIXt values, that column will be used automatically.
boundary	A single Date or POSIXt value representing the point after which points should be treated as having occurred in the second time period. See 'Details'.
groups	Name of a column in data containing exactly two unique non-missing values, which will be used to identify whether each row should be counted in the first (before) or second (after) groups. Which groups to use will be determined by calling sort(unique(groups)). If groups is not a factor, a message will be printed confirming which value has been used for which group. See 'Details'.
cell_size	numeric value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the sf data frame given in the data argument. Ignored if grid is not NULL. If this argument and grid are NULL (the default), the cell size will be calculated automatically (see Details).
grid_type	character specifying whether the grid should be made up of squares ("rect", the default) or hexagons ("hex"). Ignored if grid is not NULL.
grid	sf data frame containing points containing polygons, which will be used as the grid for which counts are made.
quiet	if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.

Details

This function creates a regular two-dimensional grid of cells (unless a custom grid is specified with `grid`) and calculates the difference between the number of points in each grid cell:

- before and after a set point in time, if `boundary` is specified,
- between two groups of points, if a column of grouping values is specified with `groups`,
- before and after the mid-point of the dates/times present in the data, if both `boundary` and `groups` are `NULL` (the default).

If both `boundary` and `groups` are not `NULL`, the value of `boundary` will be ignored.

Coverage of the output data:

The grid produced by this function covers the convex hull of the input data layer. This means the result may include zero counts for cells that are outside the area for which data were provided, which could be misleading. To handle this, consider cropping the output layer to the area for which data are available. For example, if you only have crime data for a particular district, crop the output dataset to the district boundary using `st_intersection`.

Automatic cell-size selection:

If no cell size is given then the cell size will be set so that there are 50 cells on the shorter side of the grid. If the `data` SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

Value

An `sf` tibble of regular grid cells with corresponding hot-spot classifications for each cell. This can be plotted using `autoplot`.

See Also

`hotspot_dual_kde()` for comparing the density of two layers, which will often be more useful than comparing counts if the point locations represent and underlying continuous distribution.

Examples

```
# Compare counts from the first half of the period covered by the data to
# counts from the second half

hotspot_change(memphis_robberies)


# Create a grouping variable, then compare counts across values of that
# variable

memphis_robberies$weekend <-
  weekdays(memphis_robberies$date) %in% c("Saturday", "Sunday")
hotspot_change(memphis_robberies, groups = weekend)
```

`hotspot_classify` *Classify hot-spots*

Description

Classify cells in a grid based on changes in the clustering of points (typically representing events) in a two-dimensional regular grid over time.

Usage

```
hotspot_classify(
  data,
  time = NULL,
  period = NULL,
  start = NULL,
  cell_size = NULL,
  grid_type = "rect",
  grid = NULL,
  collapse = FALSE,
  params = hotspot_classify_params(),
  quiet = FALSE
)
```

Arguments

`data`	sf data frame containing points.
`time`	Name of the column in `data` containing `Date` or `POSIXt` values representing the date associated with each point. If this argument is `NULL` and `data` contains a single column of `Date` or `POSIXt` values, that column will be used automatically.
`period`	A character value containing a number followed by a unit of time, e.g. for example, "12 months" or "3.5 days", where the unit of time is one of second, minute, hour, day, week, month, quarter or year (or their plural forms).
`start`	A `Date` or `POSIXt` value specifying when the first temporal period should start. If `NULL` (the default), the first period will start at the beginning of the earliest date found in the data (if `period` is specified in days, weeks, months, quarters or years) or at the earliest time found in the data otherwise.
`cell_size`	`numeric` value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the `sf` data frame given in the `data` argument. Ignored if `grid` is not `NULL`. If this argument and `grid` are `NULL` (the default), the cell size will be calculated automatically (see Details).
`grid_type`	`character` specifying whether the grid should be made up of squares (`"rect"`, the default) or hexagons (`"hex"`). Ignored if `grid` is not `NULL`.
`grid`	sf data frame containing points containing polygons, which will be used as the grid for which counts are made.

collapse	If the range of dates in the data is not a multiple of period, the final period will be shorter than the others. In that case, should this shorter period be collapsed into the penultimate period?
params	A list of optional parameters that can affect the output. The list can be produced most easily using the hotspot_classify_params helper function.
quiet	if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.

Value

An sf tibble of regular grid cells with corresponding hot-spot classifications for each cell. This can be plotted using autoplot.

Hot-spots are spatial areas that contain more points than would be expected by chance; cold-spots are areas that contain fewer points than would be expected. Whether an area is a hot-spot can vary over time. This function creates a space-time cube, determines whether an area is a hot-spot for each of several consecutive time periods and uses that to classify areas according to whether they are persistent, intermittent, emerging or former hot- or cold-spots.

Hot and cold spots:

Hot- and cold-spots are identified by calculating the Getis-Ord G_i^* (gi-star) or G_i Z-score statistic for each cell in a regular grid for each time period. Cells are classified as follows, using the parameters provided in the params argument:

- *Persistent hot-/cold-spots* are cells that have been hot-/cold-spots consistently over time. Formally: if the p-value is less than critical_p for at least persistent_prop proportion of time periods.
- *Emerging hot-/cold-spots* are cells that have become hot-/cold-spots recently but were not previously. Formally: if the p-value is less than critical_p for at least hotspot_prop of time periods defined as recent by recent_prop but the p-value was *not* less than critical_p for at least hotspot_prop of time periods defined as non-recent by 1 - recent_prop.
- *Former hot-/cold-spots* are cells that used to be hot-/cold-spots but have not been more recently. Formally: if the p-value was less than critical_p for at least hotspot_prop of time periods defined as non-recent by 1 - recent_prop but the p-value was *not* less than critical_p for for at least hotspot_prop of time periods defined as recent by recent_prop.
- *Intermittent hot-/cold-spots* are cells that have been hot-/cold-spots, but not as frequently as persistent hotspots and not only during recent/non-recent periods. Formally: if the p-value is less than critical_p for at least hotspot_prop of time periods but the cell is not an emerging or former hotspot.
- *No pattern* if none of the above categories apply.

Coverage of the output data:

The grid produced by this function covers the convex hull of the input data layer. This means the result may include G_i^* or G_i values for cells that are outside the area for which data were provided, which could be misleading. To handle this, consider cropping the output layer to the area for which data are available. For example, if you only have crime data for a particular district, crop the output dataset to the district boundary using st_intersection.

Automatic cell-size selection:

If no cell size is given then the cell size will be set so that there are 50 cells on the shorter side of the grid. If the `data` SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

References

Chainey, S. (2020). *Understanding Crime: Analyzing the Geography of Crime*. Redlands, CA: ESRI.

`hotspot_classify_params`

Control the parameters used to classify hotspots

Description

This function allows specification of parameters that affect the output from `hotspot_classify`.

Usage

```
hotspot_classify_params(
  hotspot_prop = 0.1,
  persistent_prop = 0.8,
  recent_prop = 0.2,
  critical_p = 0.05,
  nb_dist = NULL,
  include_self = TRUE,
  p_adjust_method = NULL
)
```

Arguments

`hotspot_prop`	A single numeric value specifying the minimum proportion of periods for which a cell must contain significant clusters of points before the cell can be classified as a hot or cold spot of any type.
`persistent_prop`	A single numeric value specifying the minimum proportion of periods for which a cell must contain significant clusters of points before the cell can be classified as a persistent hot or cold spot.
`recent_prop`	A single numeric value specifying the proportion of periods that should be treated as being recent in the classification of emerging and former hotspots.
`critical_p`	A threshold *p*-value below which values should be treated as being statistically significant.
`nb_dist`	The distance around a cell that contains the neighbours of that cell, which are used in calculating the statistic. If this argument is `NULL` (the default), `nb_dist` is set as `cell_size * sqrt(2)` so that only the cells immediately adjacent to each cell are treated as being its neighbours.

include_self — Should points in a given cell be counted as well as counts in neighbouring cells when calculating the values of G_i^* (if include_self = TRUE, the default) or G_i (if include_self = FALSE) values? You are unlikely to want to change the default value.

p_adjust_method — The method to be used to adjust *p*-values for multiple comparisons. NULL (the default) uses the default method used by p.adjust, but any of the character values in stats::p.adjust.methods may be specified.

Value

A list that can be used as the input to the params argument to hotspot_classify.

---

hotspot_count *Count points in cells in a two-dimensional grid*

Description

Count points in cells in a two-dimensional grid

Usage

```
hotspot_count(
  data,
  cell_size = NULL,
  grid_type = "rect",
  grid = NULL,
  weights = NULL,
  quiet = FALSE
)
```

Arguments

data — sf data frame containing points.

cell_size — numeric value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the sf data frame given in the data argument. Ignored if grid is not NULL. If this argument and grid are NULL (the default), the cell size will be calculated automatically (see Details).

grid_type — character specifying whether the grid should be made up of squares ("rect", the default) or hexagons ("hex"). Ignored if grid is not NULL.

grid — sf data frame containing polygons, which will be used as the grid for which counts are made.

weights — NULL or the name of a column in data to be used as weights for weighted counts.

quiet — if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.

Details

This function counts the number of points in each cell in a regular grid. If a column name in `data` is supplied with the `weights` argument, weighted counts will also be produced.

Automatic cell-size selection:

If `grid` is `NULL` and no cell size is given, the cell size will be set so that there are 50 cells on the shorter side of the grid. If the `data` SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

Value

An `sf` tibble of regular grid cells with corresponding point counts for each cell. This can be plotted using `autoplot`.

Examples

```
# Set cell size automatically

hotspot_count(memphis_robberies_jan)


# Transform data to UTM zone 15N so that cell_size and bandwidth can be set
# in metres
library(sf)
memphis_robberies_utm <- st_transform(memphis_robberies_jan, 32615)

# Manually set grid-cell size in metres, since the `memphis_robberies_utm`
# dataset uses a co-ordinate reference system (UTM zone 15 north) that is
# specified in metres

hotspot_count(memphis_robberies_utm, cell_size = 200)
```

---

hotspot_dual_kde *Estimate the relationship between the kernel density of two layers of points*

Description

Estimate the relationship between the kernel density of two layers of points

Usage

```
hotspot_dual_kde(
  x,
  y,
  cell_size = NULL,
```

```
  grid_type = "rect",
  bandwidth = NULL,
  bandwidth_adjust = 1,
  method = "ratio",
  grid = NULL,
  weights = NULL,
  quiet = FALSE,
  ...
)
```

Arguments

x, y	sf data frames containing points.
cell_size	numeric value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the sf data frame given in the x argument. Ignored if grid is not NULL. If this argument and grid are NULL (the default), the cell size will be calculated automatically (see Details).
grid_type	character specifying whether the grid should be made up of squares ("rect", the default) or hexagons ("hex"). Ignored if grid is not NULL.
bandwidth	either a single numeric value specifying the bandwidth to be used in calculating the kernel density estimates, or a list of exactly 2 such values. If this argument is NULL (the default), the bandwidth for both x and y will be determined automatically using the result of bandwidth.nrd called on the co-ordinates of the points in x. If this argument is list(NULL, NULL), separate bandwidths will be determined automatically for x and y based on each layer.
bandwidth_adjust	single positive numeric value by which the value of bandwidth for both x and y will be multiplied, or a list of two such values. Useful for setting the bandwidth relative to the default.
method	character specifying the method by which the densities, d(), of x and y will be related: ratio (the default) calculates the density of x divided by the density of y, i.e. d(x) / d(y). log calculates the natural logarithm of the density of x divided by the density of y, i.e. log(d(x) / d(y)). diff calculates the difference between the density of x and the density of y, i.e. d(x) - d(y). sum calculates the sum of the density of x and the density of y, i.e. d(x) + d(y). The result of this calculation will be returned in the kde column of the return value.
grid	sf data frame containing polygons, which will be used as the grid for which densities are estimated.
weights	NULL (the default) or a vector of length two giving either NULL or the name of a column in each of x and y to be used as weights for weighted counts and KDE values.

quiet	if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.
...	Further arguments passed to kde.

Value

An sf tibble of grid cells with corresponding point counts and dual kernel density estimates for each cell. This can be plotted using autoplot.

This function creates a regular two-dimensional grid of cells (unless a custom grid is specified with grid), calculates the density of points in each cell for each of x and y using functions from the SpatialKDE package, then produces a value representing a relation between the two densities. The count of points in each cell is also returned.

Dual kernel density values can be useful for understanding the relationship between the distributions of two sets of point locations. For example:

- The ratio between two densities representing the locations of burglaries and the locations of houses can show the distribution of the risk (incidence rate) of burglaries. The logged ratio may be useful to show relationships where one set of points has an extremely skewed distribution.
- The difference between two densities can show the change in distributions between two points in time.
- The sum of two densities can be used to estimate the total density of two types of point, e.g. the locations of occurrences of two diseases.

Coverage of the output data:

The grid produced by this function covers the convex hull of the points in x. This means the result may include KDE values for cells that are outside the area for which data were provided, which could be misleading. To handle this, consider cropping the output layer to the area for which data are available. For example, if you only have crime data for a particular district, crop the output dataset to the district boundary using st_intersection.

Automatic cell-size selection:

If no cell size is given then the cell size will be set so that there are 50 cells on the shorter side of the grid. If the x SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

References

Yin, P. (2020). Kernels and Density Estimation. *The Geographic Information Science & Technology Body of Knowledge* (1st Quarter 2020 Edition), John P. Wilson (ed.). doi:doi:10.22224/gistbok/2020.1.12

Examples

```
# See also the examples for `hotspot_kde()` for examples of how to specify
# `cell_size`, `bandwidth`, etc.

library(sf)
```

```
# Transform data to UTM zone 15N so that cell_size and bandwidth can be set
# in metres
memphis_robberies_utm <- st_transform(memphis_robberies, 32615)
memphis_population_utm <- st_transform(memphis_population, 32615)

# Calculate burglary risk based on residential population. `weights` is set
# to `c(NULL, population)` so that the robberies layer is not weighted and
# the population layer is weighted according to the number of residents in
# each census block.

hotspot_dual_kde(
  memphis_robberies_utm,
  memphis_population_utm,
  bandwidth = list(NULL, NULL),
  weights = c(NULL, population)
)
```

`hotspot_gistar` *Identify significant spatial clusters of points*

Description

Identify hotspot and coldspot locations, that is cells in a regular grid in which there are more/fewer points than would be expected if the points were distributed randomly.

Usage

```
hotspot_gistar(
  data,
  cell_size = NULL,
  grid_type = "rect",
  kde = TRUE,
  bandwidth = NULL,
  bandwidth_adjust = 1,
  grid = NULL,
  weights = NULL,
  nb_dist = NULL,
  include_self = TRUE,
  p_adjust_method = NULL,
  quiet = FALSE,
  ...
)
```

Arguments

data	sf data frame containing points.
cell_size	numeric value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the sf data frame given in the data argument. Ignored if grid is not NULL. If this argument and grid are NULL (the default), the cell size will be calculated automatically (see Details).
grid_type	character specifying whether the grid should be made up of squares ("rect", the default) or hexagons ("hex"). Ignored if grid is not NULL.
kde	TRUE (the default) or FALSE indicating whether kernel density estimates (KDE) should be produced for each grid cell.
bandwidth	numeric value specifying the bandwidth to be used in calculating the kernel density estimates. If this argument is NULL (the default), the bandwidth will be specified automatically using the mean result of bandwidth.nrd called on the x and y co-ordinates separately.
bandwidth_adjust	single positive numeric value by which the value of bandwidth is multiplied. Useful for setting the bandwidth relative to the default.
grid	sf data frame containing polygons, which will be used as the grid for which counts are made.
weights	NULL or the name of a column in data to be used as weights for weighted counts and KDE values.
nb_dist	The distance around a cell that contains the neighbours of that cell, which are used in calculating the statistic. If this argument is NULL (the default), nb_dist is set as cell_size * sqrt(2) so that only the cells immediately adjacent to each cell are treated as being its neighbours.
include_self	Should points in a given cell be counted as well as counts in neighbouring cells when calculating the values of G_i^* (if include_self = TRUE, the default) or G_i (if include_self = FALSE) values? You are unlikely to want to change the default value.
p_adjust_method	The method to be used to adjust p-values for multiple comparisons. NULL (the default) uses the default method used by p.adjust, but any of the character values in stats::p.adjust.methods may be specified.
quiet	if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.
...	Further arguments passed to kde or ignored if kde = FALSE.

Details

This function calculates the Getis-Ord G_i^* (gi-star) or G_i Z-score statistic for identifying clusters of point locations. The underlying implementation uses the localG function to calculate the Z scores and then p.adjustSP function to adjust the corresponding p-values for multiple comparison. The function also returns counts of points in each cell and (by default but optionally) kernel density estimates using the kde function.

Coverage of the output data:

The grid produced by this function covers the convex hull of the input data layer. This means the result may include G_i^* or G_i values for cells that are outside the area for which data were provided, which could be misleading. To handle this, consider cropping the output layer to the area for which data are available. For example, if you only have crime data for a particular district, crop the output dataset to the district boundary using `st_intersection`.

Automatic cell-size selection:

If no cell size is given then the cell size will be set so that there are 50 cells on the shorter side of the grid. If the `data` SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

Value

An `sf` tibble of regular grid cells with corresponding point counts, G_i or G_i^* values and (optionally) kernel density estimates for each cell. Values greater than zero indicate more points than would be expected for randomly distributed points and values less than zero indicate fewer points. Critical values of G_i and G_i^* are given in the manual page for `localG`.

The output from this function can be plotted in the same way as for other SF objects, for which see `vignette("sf5", package = "sf")`.

References

Getis, A. & Ord, J. K. (1992). The Analysis of Spatial Association by Use of Distance Statistics. *Geographical Analysis*, 24(3), 189-206. doi:doi:10.1111/j.15384632.1992.tb00261.x

Examples

```
library(sf)

# Transform data to UTM zone 15N so that cell_size and bandwidth can be set
# in metres
memphis_robberies_utm <- st_transform(memphis_robberies_jan, 32615)

# Automatically set grid-cell size, bandwidth and neighbour distance

hotspot_gistar(memphis_robberies_utm)


# Manually set grid-cell size in metres, since the `memphis_robberies`
# dataset uses a co-ordinate reference system (UTM zone 15 north) that is
# specified in metres

hotspot_gistar(memphis_robberies_utm, cell_size = 200)


# Automatically set grid-cell size and bandwidth for lon/lat data, since it
# is not intuitive to set these values manually in decimal degrees. To do
# this it is necessary to not calculate KDEs due to a limitation in the
# underlying function.
```

```
hotspot_gistar(memphis_robberies, kde = FALSE)
```

`hotspot_grid` *Create either a rectangular or hexagonal two-dimensional grid*

Description

Create either a rectangular or hexagonal two-dimensional grid

Usage

```
hotspot_grid(data, cell_size = NULL, grid_type = "rect", quiet = FALSE, ...)
```

Arguments

`data`	`sf` data frame.
`cell_size`	`numeric` value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the `sf` data frame given in the `data` argument. If this argument is `NULL` (the default), the cell size will be calculated automatically (see Details).
`grid_type`	`character` specifying whether the grid should be made up of squares (`"rect"`, the default) or hexagons (`"hex"`).
`quiet`	if set to `TRUE`, messages reporting the values of any parameters set automatically will be suppressed. The default is `FALSE`.
`...`	Further arguments passed to `link[sf]{st_make_grid}`.

Value

A simple features tibble containing polygons representing grid cells.

The grid will be based on the convex hull of `data`, expanded by a buffer of `cell_size / 2` to ensure all the points in `data` fall within the resulting grid.

hotspot_kde *Estimate two-dimensional kernel density of points*

Description

Estimate two-dimensional kernel density of points

Usage

```
hotspot_kde(
  data,
  cell_size = NULL,
  grid_type = "rect",
  bandwidth = NULL,
  bandwidth_adjust = 1,
  grid = NULL,
  weights = NULL,
  quiet = FALSE,
  ...
)
```

Arguments

data	sf data frame containing points.
cell_size	numeric value specifying the size of each equally spaced grid cell, using the same units (metres, degrees, etc.) as used in the sf data frame given in the data argument. Ignored if grid is not NULL. If this argument and grid are NULL (the default), the cell size will be calculated automatically (see Details).
grid_type	character specifying whether the grid should be made up of squares ("rect", the default) or hexagons ("hex"). Ignored if grid is not NULL.
bandwidth	numeric value specifying the bandwidth to be used in calculating the kernel density estimates. If this argument is NULL (the default), the bandwidth will be determined automatically using the result of bandwidth.nrd called on the co-ordinates of data.
bandwidth_adjust	single positive numeric value by which the value of bandwidth is multiplied. Useful for setting the bandwidth relative to the default.
grid	sf data frame containing polygons, which will be used as the grid for which densities are estimated.
weights	NULL or the name of a column in data to be used as weights for weighted counts and KDE values.
quiet	if set to TRUE, messages reporting the values of any parameters set automatically will be suppressed. The default is FALSE.
...	Further arguments passed to kde.

Details

This function creates a regular two-dimensional grid of cells (unless a custom grid is specified with `grid`) and calculates the density of points in each cell on that grid using functions from the `SpatialKDE` package. The count of points in each cell is also returned.

Coverage of the output data:

The grid produced by this function covers the convex hull of the input data layer. This means the result may include KDE values for cells that are outside the area for which data were provided, which could be misleading. To handle this, consider cropping the output layer to the area for which data are available. For example, if you only have crime data for a particular district, crop the output dataset to the district boundary using `st_intersection`.

Automatic cell-size selection:

If no cell size is given then the cell size will be set so that there are 50 cells on the shorter side of the grid. If the `data` SF object is projected in metres or feet, the number of cells will be adjusted upwards so that the cell size is a multiple of 100.

Value

An `sf` tibble of grid cells with corresponding point counts and kernel density estimates for each cell. This can be plotted using `autoplot`.

References

Yin, P. (2020). Kernels and Density Estimation. *The Geographic Information Science & Technology Body of Knowledge* (1st Quarter 2020 Edition), John P. Wilson (ed.). doi:doi:10.22224/gistbok/2020.1.12

Examples

```
library(sf)

# Transform data to UTM zone 15N so that cell_size and bandwidth can be set
# in metres
memphis_robberies_utm <- st_transform(memphis_robberies_jan, 32615)

# Automatically set grid-cell size, bandwidth and neighbour distance

hotspot_kde(memphis_robberies_utm)


# Manually set grid-cell size and bandwidth in metres, since the
# `memphis_robberies_utm` dataset uses a co-ordinate reference system (UTM
# zone 15 north) that is specified in metres

hotspot_kde(memphis_robberies_utm, cell_size = 200, bandwidth = 1000)
```

memphis_population *Populations of census blocks in Memphis in 2020*

Description

A dataset containing records of populations associated with the centroids of census blocks in Memphis, Tennessee, in 2020.

Usage

```
memphis_population
```

Format

A simple-features tibble with 10,393 rows and three variables:

geoid the census GEOID for each block

population the number of people residing in each block

geometry the co-ordinates of the centroid of each block, stored in simple-features point format

Source

US Census Bureau. Census 2020, Redistricting Data summary file. https://www.census.gov/programs-surveys/decennial-census/about/rdo/summary-files.html

memphis_precincts *Memphis Police Department Precincts*

Description

A dataset containing the boundaries of Memphis Police Department precincts.

Usage

```
memphis_precincts
```

Format

A simple-features tibble with 9 rows and two variables:

precinct the precinct name

geometry the boundary of each precinct, stored in simple-features polygon format

Licence: Public domain https://data.memphistn.gov/d/tdws-78iq

Source

City of Memphis https://data.memphistn.gov/d/rqqz-pj4u

memphis_robberies *Personal robberies in Memphis in 2019*

Description

A dataset containing records of personal robberies recorded by police in Memphis, Tennessee, in 2019.

Usage

```
memphis_robberies
```

Format

A simple-features tibble with 2,245 rows and four variables:

uid a unique identifier for each robbery

offense_type the type of crime (always 'personal robbery')

date the date and time at which the crime occurred

geometry the co-ordinates at which the crime occurred, stored in simple-features point format

Source

Crime Open Database, https://osf.io/zyaqn/

memphis_robberies_jan *Personal robberies in Memphis in January 2019*

Description

A dataset containing records of personal robberies recorded by police in Memphis, Tennessee, in January 2019. This dataset is too small for some types of analysis but is included for testing purposes.

Usage

```
memphis_robberies_jan
```

Format

A simple-features tibble with 206 rows and four variables:

uid a unique identifier for each robbery

offense_type the type of crime (always 'personal robbery')

date the date and time at which the crime occurred

geometry the co-ordinates at which the crime occurred, stored in simple-features point format

Source

Crime Open Database, https://osf.io/zyaqn/

Index