

# Package: SpatialKDE (via r-universe)

October 8, 2024

**Type** Package

**Title** Kernel Density Estimation for Spatial Data

**Version** 0.8.2

**URL** <https://jancaha.github.io/SpatialKDE/index.html>,  
<https://github.com/JanCaha/SpatialKDE>

**Description** Calculate Kernel Density Estimation (KDE) for spatial data. The algorithm is inspired by the tool 'Heatmap' from 'QGIS'. The method is described by: Hart, T., Zandbergen, P. (2014) <[doi:10.1108/PIJPSM-04-2013-0039](https://doi.org/10.1108/PIJPSM-04-2013-0039)>, Nelson, T. A., Boots, B. (2008) <[doi:10.1111/j.0906-7590.2008.05548.x](https://doi.org/10.1111/j.0906-7590.2008.05548.x)>, Chainey, S., Tompson, L., Uhlig, S.(2008) <[doi:10.1057/palgrave.sj.8350066](https://doi.org/10.1057/palgrave.sj.8350066)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**VignetteBuilder** knitr

**LinkingTo** cpp11, progress

**Imports** sf, dplyr, glue, magrittr, rlang, vctrs, methods, raster

**Suggests** tmap, sp, knitr, rmarkdown, testthat (>= 2.99.0), xml2

**Config/testthat/edition** 3

**Repository** <https://jancaha.r-universe.dev>

**RemoteUrl** <https://github.com/JanCaha/SpatialKDE>

**RemoteRef** HEAD

**RemoteSha** e266aed3b3a563bc7ad67281216cc62bea6834cd

## Contents

create_grid_rectangular . . . . .	2
create_raster . . . . .	3
kde . . . . .	4

---

create\_grid\_rectangular

*Create grid*

---

### Description

Create grid of equally spaced rectangles or hexagons. The distance between centre points in both x and y dimension is equal to `cell_size`. The function is effectively a wrapper around `st_make_grid` with a little bit of preprocessing including generation of grid only inside `st_convex_hull`.

### Usage

```
create_grid_rectangular(
  geometry,
  cell_size,
  side_offset = 0,
  only_inside = FALSE
)
```

```
create_grid_hexagonal(
  geometry,
  cell_size,
  side_offset = 0,
  only_inside = FALSE
)
```

### Arguments

<code>geometry</code>	<code>sf</code> data.frame containing geometry which should be cover by the grid.
<code>cell_size</code>	numeric specifying the distance for equally spaced centers of polygons (rectangular or hexagonal).
<code>side_offset</code>	numeric specifying the side offset, distance added to the convex hull of input geometry to generate grid for KDE. Good estimate is usually the same value as band width of KDE.
<code>only_inside</code>	logical specifying if the grid cells should be generated only inside of the geometry. Default value is FALSE.

### Value

`sf` data.frame.

### Functions

- `create_grid_rectangular()`: Create rectangular grid
- `create_grid_hexagonal()`: Create hexagonal grid

## Examples

```
library(sf)
nc <- st_read(system.file("shape/nc.shp", package = "sf")) %>% st_transform(32031)
grid <- create_grid_hexagonal(nc, cell_size = 100000)
grid <- create_grid_rectangular(nc, cell_size = 100000, only_inside = TRUE)
```

---

create_raster	<i>Create raster</i>
---------------	----------------------

---

## Description

Create raster of equally spaced cells. The distance between centre of cells in both x and y dimension is equal to `cell_size`.

## Usage

```
create_raster(geometry, cell_size, side_offset = 0)
```

## Arguments

<code>geometry</code>	<code>sf</code> data.frame containing geometry which should be cover by the raster.
<code>cell_size</code>	numeric specifying the distance for equally spaced cells.
<code>side_offset</code>	numeric specifying the side offset, distance added to the convex hull of input geometry to generate raster for KDE. Good estimate is usually the same value as band width of KDE.

## Value

[Raster-class](#)

## Examples

```
library(sf)
nc <- st_read(system.file("shape/nc.shp", package = "sf")) %>% st_transform(32031)
raster <- create_raster(nc, cell_size = 100000)
```

kde

*Kernel Density Estimation***Description**

KDE for spatial data. The algorithm is heavily inspired by [Heatmap tool](#) in QGIS. The help for QGIS tools is provided [at the QGIS website](#). The a tutorial is provided [here](#).

**Usage**

```
kde(
  points,
  band_width,
  decay = 1,
  kernel = c("quartic", "uniform", "triweight", "epanechnikov", "triangular"),
  scaled = FALSE,
  weights = c(),
  grid,
  cell_size,
  quiet = FALSE
)
```

**Arguments**

points	<a href="#">sf</a> data.frame containing only POINTS.
band_width	numeric specifying the band width for KDE.
decay	numeric specifying the decay parameter for "triangular" kernel. For other kernels besides "triangular" the parameter is not used.
kernel	character specifying type of kernel to use. Available implemented kernels are "uniform", "quartic", "triweight", "epanechnikov", "triangular". Default is "quartic" and if unknown kernel name is used it falls back to the default value.
scaled	logical specifying if the output values should be scaled. Default value is FALSE.
weights	numeric vector of weights for individual points.
grid	either <a href="#">sf</a> data.frame (outcome of function <a href="#">create_grid_rectangular</a> or <a href="#">create_grid_hexagonal</a> ) or <a href="#">Raster-class</a> (outcome of function <a href="#">create_raster</a> ). Does not have to be specified if cell_size is set.
cell_size	numeric specifying the distance for equal spaced points. Must be higher than 0. Can be left out if grid is provided as grid is used instead. The code used to generate grid is <a href="#">create_grid_rectangular</a> (points, cell_size, band_width).
quiet	Should printing of progress bar be suppressed? Default 'FALSE'.

**Details**

grid parameter specifies output of the function. KDE is calculated on the specified grid. If grid is [Raster-class](#) then outcome is also [Raster-class](#). If grid is [sf](#) data.frame then outcome is also [sf](#) data.frame.

**Value**

either [sf](#) data.frame or [Raster-class](#) depending on class of grid parameter.

**Examples**

```
library(sf)
nc <- st_read(system.file("shape/nc.shp", package = "sf")) %>% st_transform(32031)
grid <- create_grid_hexagonal(nc, cell_size = 100000)
points <- st_sample(nc, 500) %>% st_as_sf()
kde_estimate_grid <- kde(points, band_width = 150000, grid = grid)
raster <- create_raster(nc, cell_size = 100000)
kde_estimate_raster <- kde(points, band_width = 150000, grid = raster)
```

# Index

`create_grid_hexagonal`, 4  
`create_grid_hexagonal`  
    (`create_grid_rectangular`), 2  
`create_grid_rectangular`, 2, 4  
`create_raster`, 3, 4

`kde`, 4

`sf`, 2–5  
`st_convex_hull`, 2  
`st_make_grid`, 2