Package ‘buffeRs’

August 22, 2021

Title Buffer Generation for Spatial Models
Version 0.31
Date 2021-08-22
Description Generates non-circular simple feature geometries e.g. for the use as buffers in model-building.
URL https://github.com/tlhenvironment/buffeRs
Imports sf
Suggests openair, knitr, rmarkdown
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 7.0.2
VignetteBuilder knitr
NeedsCompilation no
Author Tilman Leo Hohenberger [aut, cre]
Maintainer Tilman Leo Hohenberger <tlh@ust.hk>
Depends R (>= 3.5.0)
Repository CRAN
Date/Publication 2021-08-22 08:40:12 UTC

R topics documented:

buffer_circle ................................................. 2
buffer_rectangle .............................................. 2
buffer_semicircle ............................................. 3
buffer_square ............................................... 4
buffer_wedge ............................................... 5
buffer_windrose ........................................... 5
wind_sample ............................................... 6

Index 7
buffer_circle    A circule function

Description

Creates a circular buffer. Wrapper around sf::st_buffer()

Usage

buffer_circle(point, radius, ...)

Arguments

point            Center point of the buffer, must equal to true in: sf::st_is(point,"POINT")
radius           Radius of the buffer (numeric)
...              Further arguments to give to sf::st_buffer()

Value

An object of class sfc_POLYGON

Examples

example_point = sf::st_point(c(1,2))
example_point = sf::st_sfc(example_point)
example_point = sf::st_sf(example_point)

buffer_circle(example_point, 200) -> circular_buffer
plot(circular_buffer)

buffer_rectangle    A rectangle Function

Description

Creates a rectangular polygon

Usage

buffer_rectangle(point, x_length, y_length, degree = 0)
buffer_semicircle

Arguments

point Centre point of the buffer, must equal to true in: sf::st_is(point, "POINT")

x_length Length of the x-side (horizontal side), should be in the unit of projection (numeric)

y_length Length of the y-side (vertical side), should be in the unit of projection (numeric)

degree The angle at which the rectangle is centred (clockwise). Must be between 0 and 360 (numeric)

Value

An object of class sfc_POLYGON

References

Rotation function taken from Edzer Pebesma sf package vignette https://r-spatial.github.io/sf/articles/sf3.html

Examples

example_point = sf::st_point(c(1,2))
example_point = sf::st_sfc(example_point)
example_point = sf::st_sf(example_point)
buffer_rectangle(example_point, 200, 90, 22) -> rectangular_shaped_buffer
plot(rectangular_shaped_buffer)

buffer_semicircle A Semicircle Function

Description

Creates a semicircular polygon, wrapper around buffer_wedge(point, radius, degree, degree_width = 45)

Usage

buffer_semicircle(point, radius, degree)

Arguments

point Centre point of the buffer, must equal to true in: sf::st_is(point, "POINT")

radius Radius of the buffer (numeric)

degree The angle at which the wedge is centred (clockwise). Must be between 0 and 360 (numeric)

Value

An object of class sfc_POLYGON
buffer_square

Examples

```r
example_point = sf::st_point(c(1,2))
example_point = sf::st_sfc(example_point)
example_point = sf::st_sf(example_point)

buffer_semicircle(example_point, 200, 90) -> semicircular_buffer
plot(semicircular_buffer)
```

---

**buffer_square**  
* A square Function

**Description**

Creates a square polygon, wrapper around `buffer_rectangle(point, x_length = length, y_length = length, degree)`

**Usage**

```r
buffer_square(point, length, degree = 0)
```

**Arguments**

- **point**: Centre point of the buffer, must equal to true in: `sf::st_is(point,"POINT")`
- **length**: Length of the square sides, should be in the unit of projection (numeric)
- **degree**: The angle at which the square is centred (clockwise). Must be between 0 and 360 (numeric)

**Value**

An object of class sfc_POLYGON

**Examples**

```r
example_point = sf::st_point(c(1,2))
example_point = sf::st_sfc(example_point)
example_point = sf::st_sf(example_point)
buffer_square(example_point, 90, 22) -> square_shaped_buffer
plot(square_shaped_buffer)
```
buffer_wedge

A wedge Function

Description

Creates a wedge polygon

Usage

buffer_wedge(point, radius, degree, degree_width)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>point</td>
<td>Centre point of the buffer, must equal to true in: sf::st_is(point,&quot;POINT&quot;)</td>
</tr>
<tr>
<td>radius</td>
<td>Radius of the buffer (numeric)</td>
</tr>
<tr>
<td>degree</td>
<td>The angle at which the wedge is centred (clockwise). Must be between 0 and</td>
</tr>
<tr>
<td></td>
<td>360 (numeric)</td>
</tr>
<tr>
<td>degree_width</td>
<td>Width of the wedge. Must be between 0 and 360 (numeric)</td>
</tr>
</tbody>
</table>

Value

An object of class sfc_POLYGON

Examples

element_point = sf::st_point(c(1,2))
element_point = sf::st_sfc(element_point)
element_point = sf::st_sf(element_point)
buffer_wedge(element_point, 200, 90, 45) -> wedge_shaped_buffer
plot(wedge_shaped_buffer)

buffer_windrose

A wind-rose shaped buffer function

Description

buffer_windrose creates a wind-rose based buffer shape.

Usage

buffer_windrose(point, wind_frequency_df, radius = 100, width_factor = 2)
wind_sample

**Arguments**

- **point**
  Centre point of the buffer, must equal to true in: `sf::st_is(point,"POINT")`

- **wind_frequency_df**
  A wind frequency table, in the format provided by "openair::windRose(wind_sample)$data"

- **radius**
  radius of the buffer (numeric). The radius of the largest sub-wedge of the wind-rose shaped buffer

- **width_factor**
  Scaling factor of the width of sub-wedges (numeric). Smaller number (<1) emphasize less-dominant wind-directions, (>1) emphasize dominant wind-directions.

**Value**

An object of class `sfc_POLYGON`

**Examples**

```r
example_point = sf::st_point(c(1,2))
example_point = sf::st_sfc(example_point)
example_point = sf::st_sf(example_point)

openair::windRose(wind_sample) -> wind_sample_wind_rose
wind_sample_wind_rose$data -> wind_frequency_df

buffer_windrose(example_point, wind_frequency_df, 100, 0.5) -> windrose_shaped_buffer
plot(windrose_shaped_buffer)
```

---

**wind_sample**  
*Wind Data of Hong Kong’s King’s Park meteorological station for 2010 in hourly resolution*

**Description**

Wind Data of Hong Kong’s King’s Park meteorological station for 2010 in hourly resolution

**Usage**

`wind_sample`

**Format**

Dataframe with 8549 rows and 3 variables:

- **Time**  datetime
- **ws** wind speed in m/s
- **wd** dominating wind directions

**Source**

Index

* Wind-rose
  buffer_windrose, 5
* circle
  buffer_circle, 2
* datasets
  wind_sample, 6
* rectangle
  buffer_rectangle, 2
* semicircle
  buffer_semicircle, 3
* square
  buffer_square, 4
* wedge
  buffer_semicircle, 3
  buffer_wedge, 5
* wind
  buffer_windrose, 5