Package 'pointdexter'

March 17, 2019

Type Package

Title Labels Points Inside Polygons

Date 2019-03-16

Version 0.1.1

Maintainer Cristian E. Nuno <cenuno@syr.edu>

Description Labels longitudinal and latitudinal coordinates located inside a polygon.
For a singular polygon, the label is a logical vector of TRUE and FALSE values. For multiple polygons, the label is a character vector based on the names of each polygon.
The package is designed to work with both sf and SpatialPolygonsDataFrame objects.

URL https://cenuno.github.io/pointdexter/,
https://github.com/cenuno/pointdexter

License GPL-3

Encoding UTF-8

Depends R (>= 3.3.0)

Imports sp (>= 1.3-1), splancs (>= 2.01-40), utils (>= 3.3.0)

Suggests knitr (>= 1.21), rmarkdown (>= 1.11), testthat (>= 2.0.1)

Enhances sf (>= 0.7)

VignetteBuilder knitr

RoxygenNote 6.1.1

Language en-US

BugReports https://github.com/cenuno/pointdexter/issues

NeedsCompilation no

Author Cristian E. Nuno [aut, cre]

Repository CRAN

Date/Publication 2019-03-17 06:00:03 UTC

R topics documented:

census_tracts_sf . . . 2
census_tracts_spdf . . . 3
city_boundary_sf . . . 5
city_boundary_spdf . . . 6
community_areas_sf . . . 7
community_areas_spdf . . . 8
cps_sy1819 . . . 9
GetPolygonBoundaries . . . 12
LabelPointsWithinPolygons . . . 14
pointdexter . . . 16

Index **17**

`census_tracts_sf` *A simple feature of the 2010 Chicago census tracts*

Description

A simple feature containing the 2010 Chicago census tracts that can only be used with the `sf` package.

Usage

```
data("census_tracts_sf")
```

Format

A simple feature (which is either `data.frame` or `tibble`) with a geometry list-column with 801 observations on the following 10 variables:

`statefp10` 17, a 2-digit code from the American National Standards Institute/Federal Information Processing Standards (ANSI/FIPS) that represents the State of Illinois

`name10` un-padded 3-to-6 digit (excluding the decimal point) name of the census tract

`commarea_n` numeric name of the community area that the census tract lays within, which can be used as a crosswalk to identify particular census tracts in a given community area

`namelsad10` character name of the census tract

`commarea` numeric name of the community area that the census tract lays within, which can be used as a crosswalk to identify particular census tracts in a given community area

`geoid10` 11-digit code that unique identifies the census tract as a concatenation of `statefp10` (2-digits) + `countyfp10` (3-digits) + `tractce` (6-digits)

`notes` not every census tract lays exactly within a community area which is where the City of Chicago made notes regarding instances of overlap, low population, or a census tract falling outside the City's boundaries

`tractce10` padded 6-digit name of the census tract

`countyfp10` 3-digit county name for Cook County

`geometry` a list that contains the dimensions and the simple feature geometry type

Details

The following is sf installation advice from Matt Herman:

Depending on your operating system and available libraries, sf can be tricky to install the first time. The sf website is a good place to start if you're having trouble. If you're using macOS, this is a good guide to installing the required libraries.

Source

This sf object comes from the City of Chicago Data Portal, Boundaries - Census Tracts - 2010. The file was last updated on July 11, 2018.

References

sf documentation has helpful information on how to work with sf objects.

See Also

- ?pointdexter::census_tracts_spdf

Examples

```
# load necessary packages ----
library(sf)

# load necessary data ----
data("census_tracts_sf")

# plot Chicago 2010 census tracts ----
par(mar = c(0, 0, 1, 0))

plot(census_tracts_sf$geometry
     , main = "2010 City of Chicago census tracts"
     , col = "gray85"
     , border = "dodgerblue4")
```

---

census_tracts_spdf *A SpatialPolygonsDataFrame of the 2010 Chicago census tracts*

Description

A SpatialPolygonsDataFrame containing the 2010 Chicago census tracts that can only be used with sp package.

Usage

```
data("census_tracts_spdf")
```

Format

A SpatialPolygonsDataFrame with 801 observations on the following 10 variables:

`statefp10` 17, a 2-digit code from the American National Standards Institute/Federal Information Processing Standards (ANSI/FIPS) that represents the State of Illinois

`name10` un-padded 3-to-6 digit (excluding the decimal point) name of the census tract

`commarea_n` numeric name of the community area that the census tract lays within, which can be used as a crosswalk to identify particular census tracts in a given community area

`namelsad10` character name of the census tract

`commarea` numeric name of the community area that the census tract lays within, which can be used as a crosswalk to identify particular census tracts in a given community area

`geoid10` 11-digit code that unique identifies the census tract as a concatenation of `statefp10` (2-digits) + `countyfp10` (3-digits) + `tractce` (6-digits)

`notes` not every census tract lays exactly within a community area which is where the City of Chicago made notes regarding instances of overlap, low population, or a census tract falling outside the City's boundaries

`tractce10` padded 6-digit name of the census tract

`countyfp10` 3-digit county name for Cook County

`geometry` a list that contains the dimensions and the simple feature geometry type

Source

This SpatialPolygonsDataFrame object comes from the City of Chicago Data Portal, Boundaries - Census Tracts - 2010. The file was last updated on July 11, 2018.

See Also

- `?pointdexter::census_tracts_sf`

Examples

```
# load necessary packages ----
library(sp)

# load necessary data ----
data("census_tracts_spdf")

# plot Chicago 2010 census tracts ----
par(mar = c(0, 0, 1, 0))

plot(census_tracts_spdf
     , main = "2010 City of Chicago census tracts"
     , col = "gray85"
     , border = "dodgerblue4")
```

`city_boundary_sf` *City boundary of Chicago*

Description

A simple feature containing the city boundary of Chicago that can only be used with the `sf` package.

Usage

```
data("city_boundary_sf")
```

Format

A simple feature (which is either `data.frame` or `tibble`) with a geometry list-column with 1 observation on the following 5 variables:

`name` CHICAGO

`objectid` value of 1

`shape_area` the approximate area of the polygon in square meters

`shape_len` unknown

`geometry` a list that contains the dimensions and the simple feature geometry type

Details

The following is `sf` installation advice from Matt Herman:

Depending on your operating system and available libraries, `sf` can be tricky to install the first time. The `sf` website is a good place to start if you're having trouble. If you're using macOS, this is a good guide to installing the required libraries.

Source

This `sf` object comes from the City of Chicago Data Portal, Boundaries - City. The file was last updated on June 30, 2017.

References

`sf` documentation has helpful information on how to work with `sf` objects.

See Also

- `?pointdexter::city_boundary_spdf`

Examples

```
# load necessary packages ----
library(sf)

# load necessary data ----
data("city_boundary_sf")

# plot Chicago ----
par(mar = c(0, 0, 1, 0))

plot(city_boundary_sf$geometry
     , main = "City of Chicago boundary"
     , col = "gray85"
     , border = "dodgerblue4")
```

`city_boundary_spdf` *A SpatialPolygonsDataFrame of the city boundary of Chicago*

Description

A SpatialPolygonsDataFrame of the city boundary of Chicago that can only be used with the sp package.

Usage

```
data("city_boundary_spdf")
```

Format

A SpatialPolygonsDataFrame with 1 observation on the following 4 variables:

`name` CHICAGO

`objectid` value of 1

`shape_area` the approximate area of the polygon in square meters

`shape_len` unknown

Source

This SpatialPolygonsDataFrame comes from the City of Chicago Data Portal, Boundaries - City. The file was last updated on June 30, 2017.

See Also

- `?pointdexter::city_boundary_sf`

Examples

```
# load necessary packages ----
library(sp)

# load necessary data ----
data("city_boundary_spdf")

# plot Chicago ----
par(mar = c(0, 0, 1, 0))

plot(city_boundary_spdf
     , main = "City of Chicago boundary"
     , col = "gray85"
     , border = "dodgerblue4")
```

`community_areas_sf` *Chicago's 77 community areas*

Description

A simple feature of the 77 community area boundaries in Chicago that can only be used with the `sf` package.

Usage

```
data("community_areas_sf")
```

Format

A simple feature (which is either `data.frame` or `tibble`) with a geometry list-column with 77 observations on the following 3 variables:

`community` the name of each community area

`area_numbe` the area number for each community(identical to `area_num_1`)

`geometry` a list that contains the dimensions and the simple feature geometry type

Details

The following is `sf` installation advice from Matt Herman:

Depending on your operating system and available libraries, `sf` can be tricky to install the first time. The `sf` website is a good place to start if you're having trouble. If you're using macOS, this is a good guide to installing the required libraries.

Source

This data frame comes from the City of Chicago Data Portal, Boundaries - Community Areas (current). The file was last updated on December 18, 2018.

References

sf documentation has helpful information on how to work with sf objects.

See Also

- ?pointdexter::community_areas_spdf

Examples

```
# load necessary packages ----
library(sf)

# load necessary data ----
data("community_areas_sf")

# plot all 77 community areas -----
par(mar = c(0, 0, 1, 0))

plot(community_areas_sf$geometry
     , main = "Chicago's 77 community areas"
     , col = "gray85"
     , border = "dodgerblue4")
```

---

community_areas_spdf *A SpatialPolygonsDataFrame of Chicago's 77 community areas*

Description

77 community area boundaries in Chicago that only be used with the sp package.

Usage

```
data("community_areas_spdf")
```

Format

A SpatialPolygonsDataFrame with 77 observations on the following 2 variables:

community the name of each community area

area_numbe the area number for each community(identical to area_num_1)

Source

This SpatialPolygonsDataFrame comes from the City of Chicago Data Portal, Boundaries - Community Areas (current). The file was last updated on December 18, 2018.

See Also

- ?pointdexter::community_areas_sf

Examples

```
# load necessary packages ----
library(sp)

# load necessary data ----
data("community_areas_spdf")

# plot all 77 community areas -----
par(mar = c(0, 0, 1, 0))

plot(community_areas_spdf
     , main = "Chicago's 77 community areas"
     , col = "gray85"
     , border = "dodgerblue4")
```

---

cps_sy1819 *Chicago Public Schools (CPS) - School Profile Information, School Year (SY) 2018-2019*

Description

School profile information for all schools in the Chicago Public School district for the school year 2018-2019.

Usage

```
data("cps_sy1819")
```

Format

A data frame of 660 observations - one record per school - on the following 81 variables:

`school_id` current 6-digit unique ID assigned to each school

`legacy_unit_id` outdated 4-digit unique ID assigned to each school

`finance_id` 5-digit unique ID assigned to each school (note that Englewood STEM High School has a value of 0 due to the fact that it is a new school set to open in the fall of 2019)

`short_name` nickname of each school

`long_name` formal name of each school

`primary_category` primary grade levels served (ES for elementary, MS for middle, and HS for high school). Note that a school can possess any of these three values and still serve students the other two values.

`is_high_school` T/F if school serves any 9-12th graders

`is_middle_school` T/F if school serves any 6-8th graders

`is_elementary_school` T/F if school serves any Kindergarten-5th graders

`is_pre_school` T/F if school serves Pre-Kindergarteners

`summary` text description of the school's mission and values

`administrator_title` either Principal or Director

`administrator` first and last name of school's administrator

`address` school's address

`city` city the school resides in (all Chicago)

`state` state the school resides in (all Illinois)

`zip` 5-digit postal code used by the United States Postal Services

`phone` 10-digit phone number

`fax` 10-digit fax number used for telephonic transmission of scanned printed material

`cps_school_profile` URL to a CPS provided web-page that provides an overview of the school

`website` URL to school's home page

`facebook` URL to school's Facebook page

`twitter` URL to school's Twitter page

`youtube` URL to a school's YouTube page

`pinterest` URL to school's Pinterest page

`attendance_boundaries` T/F if school has attendance boundaries (type in address here to see an example attendance boundary.)

`grades_offered_all` grades offered by school, separated by a comma (i.e. 1,2,3)

`grades_offered` grades offered by school, shortened by a dash (i.e. 1-3)

`student_count_total` total number of students

`student_count_low_income` number of low income students

`student_count_special_ed` total number of students

`student_count_english_learners` number of low income students

`student_count_black` total number of African-American students

`student_count_hispanic` total number of Hispanic students

`student_count_white` total number of White students

`student_count_asian` total number of Asian students

`student_count_native_american` total number of Native-American students

`student_count_other_ethnicity` total number of students whose race/ethnicity is 'Other'

`student_count_asian_pacific_islander` total number of Asian/Pacific-Islander students

`student_count_multi` total number of multi-racial/ethnic students

`student_count_hawaiian_pacific_islander` total number of Hawaiian/Pacific-Islander students

`student_count_ethnicity_not_available` total number of students whose ethnicity was not available

`statistics_description` text description about the total student count, percent low income, percent diverse learners, and percent limited english learners

`demographic_description` text description stating both the largest and second largest racial/ethnic demographic group

`dress_code` T/F if school institutes a dress code

`prek_school_day` pre-k school day description

`kindergarten_school_day` kindergarten school days are all full days

`school_hours` start and end time of school day (not a datetime)

`freshman_start_end_time` start and end time of for freshman (not a datetime)

`after_school_hours` start and end time of after school hours (not a datetime)

`earliest_drop_off_time` earliest drop off time (not a datetime)

`classroom_languages` non-English languages spoken in classroom, separated by a comma

`bilingual_services` T/F if school offers bilingual services

`refugee_services` T/F if school offers refugee services

`title_1_eligible` T/F if school is eligible for Title 1 federal funds

`preschool_inclusive` T/F if pre-school is inclusive

`preschool_instructional` T/F if pre-school is instructional

`significantly_modified` T/F if school offers services that provide students who need focused and very specific medical issues to supports for different learning styles and difficulties

`hard_of_hearing` T/F if school offers curriculum for deaf and hard-of-hearing (HOH) students

`visual_impairments` T/F if school offers curriculum for visually impaired students

`transportation_bus` CTA bus routes near the school (only route numbers)

`transportation_el` CTA 'L' train lines near the school (only the color(s) and not the exact stations)

`transportation_metra` Metra train routes near the school

`school_latitude` east-wide direction of the school (ranges from 41.65 to 42.02)

`school_longitude` north-south direction of the school (ranges from -87.84 to -87.52)

`college_enrollment_rate_school` college enrollment rates reflect the percent of CPS graduates who enrolled in college by the fall of their graduating year. CPS receives college enrollment data from the National Student Clearinghouse (NSC)

`college_enrollment_rate_mean` the mean college enrollment rate is 68.2 percent

`graduation_rate_school` high school graduation rate

`graduation_rate_mean` the mean high school graduation rate is 78.2 percent

`overall_rating` school quality rating policy (SQRP) rating, with Level 1+ being the highest performance and Level 3 being the lowest

`rating_status` a school's status determines who has decision-making power at the school level

`rating_statement` text description of the value in `overall_rating`

`classification_type` 12 different types of school admission processes and curriculum

`classification_description` text description of `classification_type`

`school_year` School Year 2018-2019

`network` District-run schools in CPS are organized into geographic networks, which provide administrative support, strategic direction, and leadership development to the schools within each network

`is_gocps_participant` GoCPS allows families to learn, research, explore, and apply to nearly every CPS school and program through one online platform

`is_gocps_prek` T/F if school serves Pre-Kindergarteners and is on GoCPS

`is_gocps_elementary` T/F if school serves elementary grade students and is on GoCPS

`is_gocps_high_school` T/F if school serves high school students and is on GoCPS

`open_for_enrollment_date` open enrollment date

Details

This data frame does not contain all columns from the source file. The following columns were dropped due to irrelevance or all schools containing NA values:

`average_act_school`, `mean_act`, `x_contact_name`, `x_contact_title`, and `closed_for_enrollment_date`.

Source

This data frame comes from the City of Chicago Data Portal, Chicago Public Schools - School Profile Information SY1819. The file was last updated on December 29, 2018.

Examples

```
# load the necessary data -----
data("cps_sy1819")
```

GetPolygonBoundaries *Obtains the boundaries of the polygon(s)*

Description

`GetPolygonBoundaries()` returns the longitudinal and latitudinal points - coordinate pairs - that make up the boundary of the polygon.

Usage

```
GetPolygonBoundaries(my.polygon, labels)
```

Arguments

`my.polygon`	Either a SpatialPolygonsDataFrame data or a sf object. See `?sp::'SpatialPolygonsDataFrame-class'` or `help(package = "sf")` for more help.
`labels`	A character vector of polygon boundary labels used to name each matrix

Details

`my.polygon` accepts a spatial object that contains a singular polygon (i.e. the boundary of the City of Chicago) or many polygons (i.e. a polygon for each the 77 Chicago community areas).

Value

If `my.polygon` is of length 1, a matrix of coordinate pairs will be returned; otherwise, a list of labeled matrices, with each matrix representing the coordinate pairs that make the boundary of each particular polygon in `my.polygon`.

See Also

- `?pointdexter::city_boundary_spdf`
- `?pointdexter::city_boundary_sf`
- `?pointdexter::community_areas_spdf`
- `?pointdexter::community_areas_sf`

Examples

```
## SpatialPolygonsDataFrame, one polygon example ## -----

# load necessary data ----
data("city_boundary_spdf")

# obtain boundaries for the City of Chicago ----
boundaries <-
  GetPolygonBoundaries(my.polygon = city_boundary_spdf)


## SpatialPolygonsDataFrame, multipolygon polygon example ## -----

# load necessary data ----
data("community_areas_spdf")

# obtain boundaries for each of the 77 Chicago community areas ----
boundaries <-
  GetPolygonBoundaries(my.polygon = community_areas_spdf
                       , labels = community_areas_spdf$community)

## sf, one polygon example ## -----

# load necessary package ----
library(sf)

# load necessary data ----
data("city_boundary_sf")

# obtain boundaries for the City of Chicago ----
boundaries <- GetPolygonBoundaries(my.polygon = city_boundary_sf)
```

```
## sf, multipolygon example ## -----

# load necessary package ----
library(sf)

# load necessary data ----
data("community_areas_sf")

# obtain boundaries for each of the 77 Chicago community areas ----
boundaries <-
  GetPolygonBoundaries(my.polygon = community_areas_sf
                       , labels = community_areas_sf$community)
```

`LabelPointsWithinPolygons` *Labels points located inside a polygon*

Description

`LabelPointsWithinPolygons()` identifies which points lie inside each polygon and labels them accordingly.

Usage

```
LabelPointsWithinPolygons(lng, lat, polygon.boundaries)
```

Arguments

`lng`	A numeric vector of non-NA longitudinal points.
`lat`	A numeric vector of non-NA latitudinal points.
`polygon.boundaries`	Either a coordinate pair matrix or a named list of coordinate pair matrices obtained from `GetPolygonBoundaries`.

Details

If points fall exactly on polygon boundaries, the default NULL gives arbitrary assignments. For more information, please see the bound argument within `splancs::inpip()`.

Value

If `polygon.boundaries` is a coordinate pair matrix, a logical vector will be returned identifying those points which lie in the polygon; otherwise, a character vector will be returned identifying the polygon each coordinate pair lies in.

See Also

- ?pointdexter::GetPolygonBoundaries
- ?pointdexter::cps_sy1819

Examples

```
## SpatialPolygonsDataFrame, one polygon example ## -----

# load necessary data ----
data("city_boundary_spdf")
data("cps_sy1819")

# obtain boundaries for the City of Chicago ----
boundaries <-
  GetPolygonBoundaries(my.polygon = city_boundary_spdf)

# identify which schools lie within the City of Chicago -----
cps_sy1819$citywide <-
  LabelPointsWithinPolygons(lng = cps_sy1819$school_longitude
                            , lat = cps_sy1819$school_latitude
                            , polygon.boundaries = boundaries)

## SpatialPolygonsDataFrame, multipolygon polygon example ## -----

# load necessary data ----
data("community_areas_spdf")
data("cps_sy1819")

# obtain boundaries for each of the 77 Chicago community areas ----
boundaries <-
  GetPolygonBoundaries(my.polygon = community_areas_spdf
                       , labels = community_areas_spdf$community)

# identify the schools which lie within each of the 77 Chicago community areas -----
cps_sy1819$community <-
  LabelPointsWithinPolygons(lng = cps_sy1819$school_longitude
                            , lat = cps_sy1819$school_latitude
                            , polygon.boundaries = boundaries)


## sf, one polygon example ## -----

# load necessary package ----
library(sf)

# load necessary data ----
data("city_boundary_sf")
data("cps_sy1819")

# obtain boundaries for the City of Chicago ----
boundaries <-
  GetPolygonBoundaries(my.polygon = city_boundary_sf)
```

```
# identify which schools lie within the City of Chicago -----
cps_sy1819$citywide <-
  LabelPointsWithinPolygons(lng = cps_sy1819$school_longitude
                            , lat = cps_sy1819$school_latitude
                            , polygon.boundaries = boundaries)



## sf, multipolygon polygon example ## -----

# load necessary package ----
library(sf)

# load necessary data ----
data("community_areas_sf")
data("cps_sy1819")

# obtain boundaries for each of the 77 Chicago community areas ----
boundaries <-
  GetPolygonBoundaries(my.polygon = community_areas_sf
                       , labels = community_areas_sf$community)


# identify the schools which lie within each of the 77 Chicago community areas -----
cps_sy1819$community <-
  LabelPointsWithinPolygons(lng = cps_sy1819$school_longitude
                            , lat = cps_sy1819$school_latitude
, polygon.boundaries = boundaries)
```

`pointdexter` *pointdexter: labels longitudinal and latitudinal coordinates located inside a polygon*

Description

The pointdexter package labels coordinate pairs located inside a polygon. For a singular polygon, the label is a logical vector of TRUE and FALSE values. For multiple polygons, the label is a character vector based on the names of each polygon. The package is designed to work with both `sf` and `SpatialPolygonsDataFrame` objects.

Details

To learn more about pointdexter, start with the vignette: `browseVignettes(package = "pointdexter")`

pointdexter functions

`GetPolygonBoundaries()` obtains the boundaries of the polygon(s)

`LabelPointsWithinPolygons()` labels points located a polygon

Index

∗Topic **datasets**
census_tracts_sf, 2
census_tracts_spdf, 3
city_boundary_sf, 5
city_boundary_spdf, 6
community_areas_sf, 7
community_areas_spdf, 8
cps_sy1819, 9

census_tracts_sf, 2
census_tracts_spdf, 3
city_boundary_sf, 5
city_boundary_spdf, 6
community_areas_sf, 7
community_areas_spdf, 8
cps_sy1819, 9

GetPolygonBoundaries, 12, *14*

LabelPointsWithinPolygons, 14

pointdexter, 16
pointdexter-package *(*pointdexter*)*, 16