HW1: Manipulating and visualizing spatio temporal data

STAT 574E: Environmental Statistics

DUE: 9/13 11:59pm

```r
##loading in libraries
library(ggplot2)
library(ggmap)
library(sf)
library(tigris)
rappdirs::user_cache_dir("tigris")
```

```
## [1] "C:\\Users\\tsgil\\AppData\\Local/tigris/tigris/Cache"
```

```r
options(tigris_use_cache = TRUE)
```

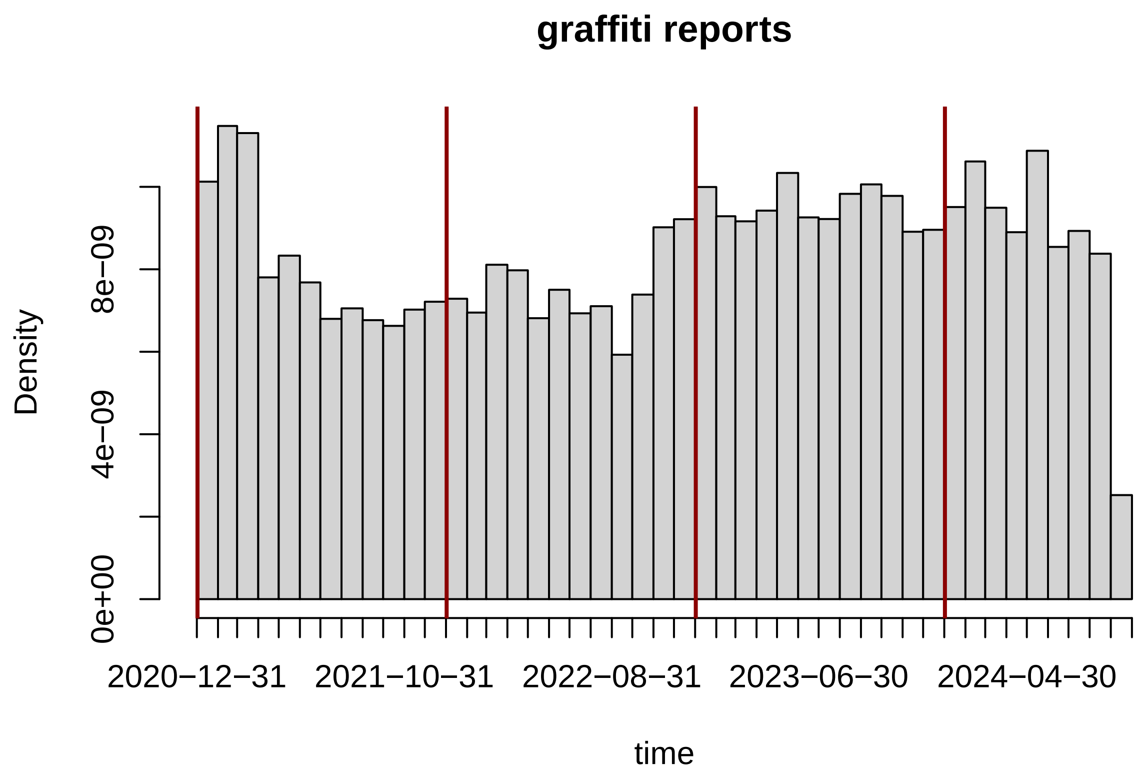
I. Time

Let's continue to look at the graffiti dataset from the city of San Diego. We'll focus on the records from 2021 onward. Here is how to read in the data and format the date as well as get the dates we want.

```r
# reading in data
data_url <- url(paste0("https://seshat.datasd.org/get_it_done_graffiti/",
                       "get_it_done_graffiti_requests_datasd.csv"))
graffiti <- read.csv(data_url)
# look at data
#View(graffiti)
Sys.sleep(2)
# setting date format
graffiti$POSIX_requested <-
  strptime(graffiti$date_requested, format = "%Y-%m-%dT%T", tz = "America/Los_Angeles")
# set date range with 'start date'
start_date <- as.POSIXlt("2021-01-01 00:00:00", tz = "America/Los_Angeles")
graffiti <- graffiti[graffiti$POSIX_requested >= start_date, ]
```

Histogram Graph colors avaliable -> black, blue, brown, cyan, darkgray, gray, green, lightgray, lime, magenta, olive, orange, pink, purple, red, teal, violet, white, yellow

```r
hist(graffiti$POSIX_requested, breaks = "month", xlab = "time", main = "graffiti reports")
abline(v = seq(as.POSIXlt("2021-01-01"), as.POSIXlt("2024-01-01"), "year"),
       col = "darkred", lwd = 2)
```



II. Simple features

A. Projections

(1) Examples of map projections: https://xkcd.com/977/.

Let's practice projecting data using the census tracts we looked at in class. We'll start by downloading those census tracts using the `tigris` package.

```r
sd_tracts <- tracts(state = "CA", county = "San Diego")
#View(sd_tracts)
```

Check to see what the current coordinate system for the `SpatialPolygonDataFrame` is with the `st_crs()` function in `sf`.

```r
st_crs(sd_tracts)
```

```
## Coordinate Reference System:
##   User input: NAD83
##   wkt:
## GEOGCRS["NAD83",
##     DATUM["North American Datum 1983",
##         ELLIPSOID["GRS 1980",6378137,298.257222101,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["latitude",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["longitude",east,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433]],
##     ID["EPSG",4269]]
```

```
cor_sd_t <- st_crs(sd_tracts)
cor_sd_t$input
```

```
## [1] "NAD83"
```

(2) Function `st_area()` can be used to compute the areas of the census tracts in `sd_tracts`.

```
# areas for each entry
areas <- st_area(sd_tracts)
# checking range
range(areas)
```

```
## Units: [m^2]
## [1]     222066.4 1636511530.0
```

The areas we just calculated use a default method for spatially-referenced data in lat/long coordinates. Now we'll try re-projecting the census tracts for San Diego to a new coordinate reference system and see how the calculated areas change.

(3) Use the `st_transform()` function to transform the census tract polygons to the **Universal Transverse Mercator (UTM) projection for zone 11** (*hint: one way to specify the CRS for UTM zone 11 is* `+proj=utm +zone=11 +datum=WGS84 +units=m +no_defs +type=crs` *but there are others*). Call the new transformed object `sd_tracts_utm`. Compute the areas for each tract in the new coordinate system.

```
#st_crs("NAD83")
# Define the target CRS for zone 11
target_crs <- st_crs(26911)
sd_tracts_utm <- st_transform(x = sd_tracts, crs = target_crs)
```

(4) One of the columns of data provided by the City of San Diego in `sd_tracts` is labeled `ALAND` for area of land. Compare the values of `ALAND` to the ones we just calculated ourselves. How well do they agree? What do you think might be a reason for discrepancies between the areas?

```
range(st_area(sd_tracts_utm))
```

```
## Units: [m^2]
## [1]     221772.9 1634555786.5
```

```r
range(sd_tracts_utm$ALAND)
```

```
## [1]          0 1635671050
```

```r
#View(sd_tracts)

# The lower values are 0 because they are in the water
# max.aland <- max(sd_tracts_utm$ALAND)
# find.max<- grep(max.aland, sd_tracts_utm$ALAND)
# max.data<- sd_tracts_utm[find.max,]
#View(max.data)
#(max.data$ALAND+max.data$AWATER)- as.numeric(st_area(max.data$geometry))

# The lower values are 0 because they are in the water
#max.aland <- max(sd_tracts$ALAND)
#find.max<- grep(max.aland, sd_tracts$ALAND)
#max.data<- sd_tracts[find.max,]
#View(max.data)
#(max.data$ALAND+max.data$AWATER)- as.numeric(st_area(max.data$geometry))
```

B. Art density

Now let's look at a new dataset from the City of San Diego. Line 16 downloads and reads into `R` the locations of artwork in the Civic Art Collection.

```r
art <- read.csv(url("https://seshat.datasd.org/civic_art_collection/public_art_locations_datasd.csv"))
art_sf <- st_as_sf(art, coords = c("lng","lat"))
st_crs(art_sf) <- st_crs(sd_tracts)
#st_crs(art_sf)
```

Suppose we're working for the Chief Operating Officer of the City of San Diego, and we're trying to decide where we should spend money on a new mural. There are two candidate locations: College-Roland Library (longitude -117.0561, latitude 32.76941), and Mission Valley Library (-117.1269, 32.7793). One way to help us decide where to commission a new mural might be to look at how much access our residents have to art in their neighborhoods. If there is a scarcity of art in the vicinity of one of these libraries, that might be a good place to consider.

First, we need to figure out how many pieces of art exist near these libraries. One way to summarize that information would be to figure out which census tract each library falls in, and then add up the number of art pieces in that census tract. We can use the function `st_intersection()` to do this.

The code in lines 19–20 creates a simple features object called `CRL` that represents the location of the College-Rolando Library (CRL).

```r
# location point for CRL
CRL <- st_as_sf(x = data.frame(lng = -117.0561, lat = 32.76941), coords = c("lng", "lat"))
st_crs(CRL) <- st_crs(sd_tracts) ## match the coordinate reference systems
#st_crs(CRL)
```

The code in line 21 finds the row in `sd_tracts` that corresponds to the census tract containing the CRL. From the output, we can see that the name of the tract is `29.05`.

```r
st_intersection(CRL, sd_tracts)
```

```
## Simple feature collection with 1 feature and 12 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -117.0561 ymin: 32.76941 xmax: -117.0561 ymax: 32.76941
## Geodetic CRS:  NAD83
##   STATEFP COUNTYFP TRACTCE       GEOID  NAME           NAMELSAD MTFCC FUNCSTAT
## 1      06      073  002905 06073002905 29.05 Census Tract 29.05 G5020        S
##     ALAND AWATER    INTPTLAT     INTPTLON                   geometry
## 1 1146262      0 +32.7711434 -117.0496451 POINT (-117.0561 32.76941)
```

(5) Fill in the blanks below to create a new variable in the `sd_tracts` dataframe called `n_art` that gives the number of art installations in each census tract. Verify you get 12 installations in the tract with the College-Rolando Library.

```r
sd_tracts$n_art <- lengths(st_intersects(sd_tracts, art_sf))
sd_tracts$n_art[sd_tracts$NAME == "29.05"]
```

```
## [1] 12
```

III. ggmap

The package ggmap is an R package the facilitates the use of freely available map tiles from sources like Google Maps, and Stadia Maps. These can provide great contextual visualizations for spatio-temporal data. Both of these maptile services require registration, but they both have free subscription levels that provide all the functionality we'll need. *Stadia does not require you to provide payment information.* Once you have registered, input your key using `ggmap::register_stadiamaps()`. Specifying `write = TRUE` will save your key for future instances of R. If you encounter problems with the CRAN distribution of ggmap, give the development version a try using `devtools::install_github("dkahle/ggmap")`.

```
## i Replacing old key (Token) with new key in C:/Users/tsgil/OneDrive/Documents/.Renviron
```

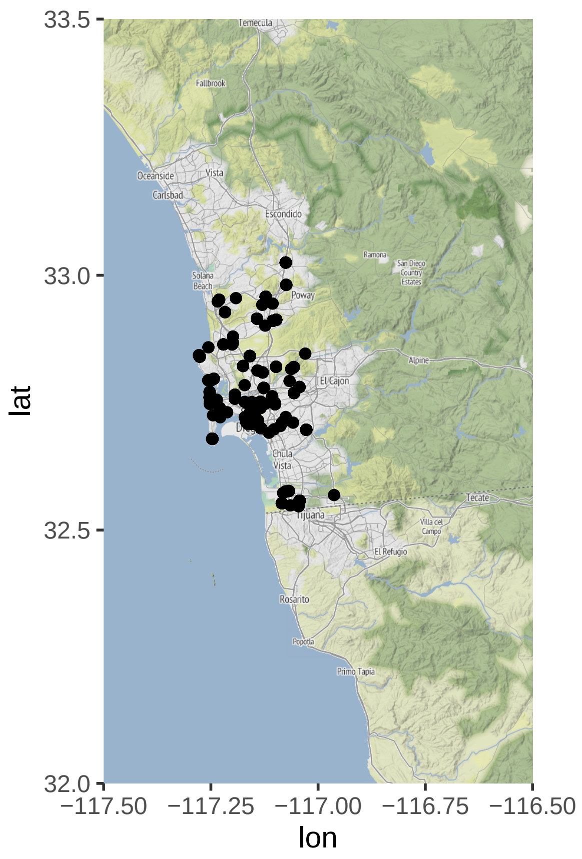
The function `get_stadiamap()` is used to download tiles from Stadia.

```r
art_map <- get_stadiamap(bbox = bbox)
```

```
## i © Stadia Maps © Stamen Design © OpenMapTiles © OpenStreetMap contributors.
```

ggmap plot

```r
#p<-
  ggmap(art_map) +
  geom_point(aes(x = lng, y = lat), data = data.frame(art))
```



```r
#print(p)
```

(6) Explore some other combinations of `maptype` and `color`. Include your maps. Which one(s) is(are) your favorite(s)?

```r
#file.show("HW1_students_1.pdf")

# For Windows
shell.exec("HW1_students_1.pdf")

# For macOS
# system("open your_file.pdf")
```